THE UNIVERSITY *of* EDINBURGH
School of Biological Sciences

Performance Improvements on LIDAEUS through Comparison with AutoDock Vina in Virtual Screening Evaluation

Student Exam Number: B134006

In partial fulfilment of the requirement for the Degree of Master of Science in Drug discovery & Translational biology at the University of Edinburgh

2018 / 2019

Name of Dissertation Supervisor: Dr. Paul Taylor

Table of Contents

Abbreviations

AUC – Area under the curve

BEDROC – Boltzmann-enhanced discrimination of receiver operating characteristic

EF – Enrichment factor

HBA – Hydrogen bond acceptor

HBD – Hydrogen bond donor

LBVS – Ligand-based virtual screening

ROC – Receiver operating characteristics

SBVS – Structure-based virtual screening

vDW – van der Waals

VS – Virtual screening

Abstract

As virtual screening has been intensely employed in early-stage drug discovery, the reliability of virtual screening programs is increasingly significant, promoting the presence of the performance evaluation of virtual screening tools as well as associated metrics. While LIDAEUS, an in-house virtual screening program, has been used in various drug discovery projects, leading to numerous successful lead compound identifications, no published research has reported its performance in a controlled system, where the number of active and inactive compounds is certain.

Therefore, we evaluated the virtual screening performance of both LIDAEUS and AutoDock Vina against 81 targets in order to compare their abilities to address the “early recognition” problem specific to virtual screening, along with the investigation into the applicability of multiple metrics and possible improvements on LIDAEUS. Further exploration of the potential of LDIAEUS was conducted by clustering the binding sites of the targets in our project based on different criteria in order to find possible target families where LIDAEUS display consistent and dominant performance.

We found that most metrics used in our project are intimately related and none of them can measure the performance of virtual screening program facing real-world problems. However, a great gap of performance between LIDAEUS and Vina was indeed revealed. While no correlation was showed between binding sites clustering and the performance of LIDAEUS, with exhaustive attempts, we succeeded in improving LIDAEUS’s performance to Vina’s level through parameterization of the posing algorithm and scoring function in LIDAEUS. Unfortunately, given the diversity of the target structures, these target-biased improvements may only exhibit in this research because of the limited amount of structures used in scoring function and posing algorithm re-parameterization. Still, our project provided an exhaustive guidance on employing LIDAEUS in research with its maximum performance.

Introduction

The first part of this work will review the history and current progress of both virtual screening and associated techniques in drug discovery. Major computational tools employed in our project will also be covered. Further, the challenges faced by virtual screening will be outlined, as they are also the problems our research attempted to address. Ultimately, the core activity of our research, the evaluation of a virtual screening program, will be reviewed in depth to reveal its significance and the limitations.

Overview of virtual screening in early-stage drug discovery

Early-stage drug discovery is a process evaluating a series of small molecules or macromolecules that can possibly be lead drug candidates (Sinha & Vohora, 2018). Given the tremendous number of chemical compounds and finite resources, efforts have been dedicated toward the development of more effective and less costly techniques to identify drug-like compounds in enormous compound libraries, which promoted the advent of high-throughput screening (HTS). Although HTS has accelerated early-stage drug discovery in the aspects of large-scale synthesis and screening of compounds, it has not greatly affected the quality of the results in a positive way. In many cases, the resulting compounds have low hit rates and further optimization can be difficult (Song et al., 2009). It is reported that the majority of these failures were due to the deficiencies in absorption, distribution, metabolism, excretion and toxicity (ADME/tox) (Lavecchia & Giovanni, 2013). Because of the above drawbacks and the remaining problems of HTS, namely the high cost of both resources and time and the uncertain mechanism of ligand-receptor binding, researchers have returned to rational and structure-based drug design (SBDD) *in silico* (Lionta et al., 2014). In contrast to previous strategies of early-stage drug discovery, SBDD is more capable of providing insight to known protein-ligand interactions, allowing highly accurate chemical modifications to ligand scaffolds (Lounnas et al., 2013). Among the methods of SBDD, virtual screening (VS) has enjoyed an

irreplaceable position in the early-stage drug discovery, which is due to its short development time and low financial cost (Sun et al., 2016).

VS, as a complement to HTS, is a collective description of the techniques that aim at identifying novel bioactive molecules in vast chemical libraries with the use of computational algorithms and models (Schneider, 2010). The resulting active compounds will be used as initial templates for further enhancement or clustered into groups by their similarities in order to establish structure-activity relationships (Matter & Sotriffer, 2011). With the application of massively parallel processing, large-scale screening can be conducted at an unprecedented speed (Taylor et al., 2008). Concretely, Lyu et al. (2019) screened 138 million library molecules with 1,500 cores in approximately 1.2 calendar days, whereas the advanced version of HTS, the ultra high-throughput screening (UHTS), can only perform 100,000 *in vitro* activity assays per day (Martis et al, 2011) and additional time of synthesis should be taken into account. In order to validate the VS results, compounds of desired activity, selected by VS, will be screened in the real world by HTS. Therefore, the libraries that VS used should consist of compounds already synthesised by the chemical vendors or applicable to synthesis. With the incomparable speed of VS, many inactive ligands from large compound libraries are not tested in real-world tests, thus lowering the cost of resources.

In silico screening can be conducted by using numerous approaches or any combinations of them, which have been traditionally categorized into two classes, namely the ligand-based and structure-based methods (Tian et al., 2013).

Ligand-based virtual screening (LBVS)

LBVS mainly utilizes structural information of known ligands with targeted activity as a probe to evaluate the probability of other molecules being active against the same macromolecule in ligand libraries, which, in other words, is measuring the molecular similarity. These structural data can be extracted

from reference molecules' specific substructures, such as the number of aromatic rings or the number of atoms or groups with the same or similar functions, which is referred to functional-class fingerprints (FCFP). With these data, LBVS is implemented by employing variable similarity-based methods, including pharmacophores, shape-based similarity, fingerprint similarity and machine learning approaches (Cereto-Massagué et al., 2014). Although scaffold hopping is often applied at the end of LBVS in order to explore diverse novel chemotypes, which is an idea about comparing molecules with different base scaffolds using data from mentioned ligand-based techniques (Böhm et al., 2004), bias introduced by the reference molecules still limits the diversity of results and is difficult to eliminate (Drwal & Griffith, 2013). Therefore, LBVS is often applied to pre-screen the libraries for size minimization before using other VS techniques (Ripphausen et al., 2010).

Structure-based virtual screening (SBVS)

Unlike LBVS bound by the input compounds information, SBVS is more likely to find completely new lead compounds according to the structural information of targets (Sun, 2016). Consequently, the quality of three-dimensional target structures is vital to SBVS. Thus, owing to the increasing number of high-quality target structures, SBVS has enjoyed a dominant position in early-stage drug discovery and has exceled in lead drug identification (Zheng et al., 2015; Xia et al., 2017). In SBVS, efforts have been made to address two problems, namely, in what position and how well a ligand can interact with the macromolecule, which are known as docking and scoring (**Figure 1**).

Figure 1. Typical workflow of SBVS.

Docking in SBVS attempts to simulate the interaction between ligands and targets using their structural data as input. Targets' structures can be derived from X-ray crystallography, nuclear magnetic resonance (NMR) and

computational methods including homology modelling and molecular dynamics (Amaro et al., 2018), which may often be processed under different requirements of varied docking programs (Cheng et al., 2012). Compounds from ligand libraries will subsequently be posed and docked into the binding site, the fitness of which will be evaluated by the scoring function (SF) using knowledge-based, force-field-based, or empirical methods. Knowledge-based SFs evaluate the fitness of docked protein–ligand binding by counting the occurrence of favoured protein-ligand interactions derived by observing known crystallographic structures (Neudert & Klebe, 2011). Such interactions are defined by the pairwise atom types within a given distance in the ligand-receptor complexes. While knowledge-based SFs are statistic-based methods, force-field-based SFs are physics-based approaches to estimating the binding affinities by summing up energy terms, including van der Waalss, electrostatic and hydrogen bonding, where classical mechanics can be applied. Thereby, such methods fail to predict the accurate entropic contributions and the desolvation energies because of the absence of reasonable physical models (Ferreira et al., 2015). Finally, empirical SFs are the approximations of binding free energy using a linear combination of all weighted energy contributions, namely, hydrophobic, hydrogen bonding and ionic interactions, entropic and desolvation effects (Liu & Wang, 2015). This empirical function is parameterized by multiple linear regressions using experimental data, due to which the performance of empirical SFs is strongly dependent on the accuracy of the training data (Ferreira et al., 2015).

In the following section, detailed docking algorithms and SFs will be illustrated with two virtual screening tools employed in this project.

LIDAEUS, LIgand Discovery at Edinburgh UniverSity, is an in-house rigid-docking program consisting of four independent modules, molecule defining, ligand posing, scoring and sorting, notably, the docking and SF employing grid-based methods (Wu et al., 2003; Taylor et al., 2008). A group of three-dimensional sitepoints are generated in a pre-defined binding pocket of the target protein according to user-customized ligand-protein interactions. Meanwhile, the hydrophobic and hydrogen bonding energies are assigned to specific types of atoms in the protein (Taylor et al., 2008). Sitepoints are coloured and represent a series of associated atoms (**Figure 2**) where ligand is mapped rigidly in order to explore the orientations.

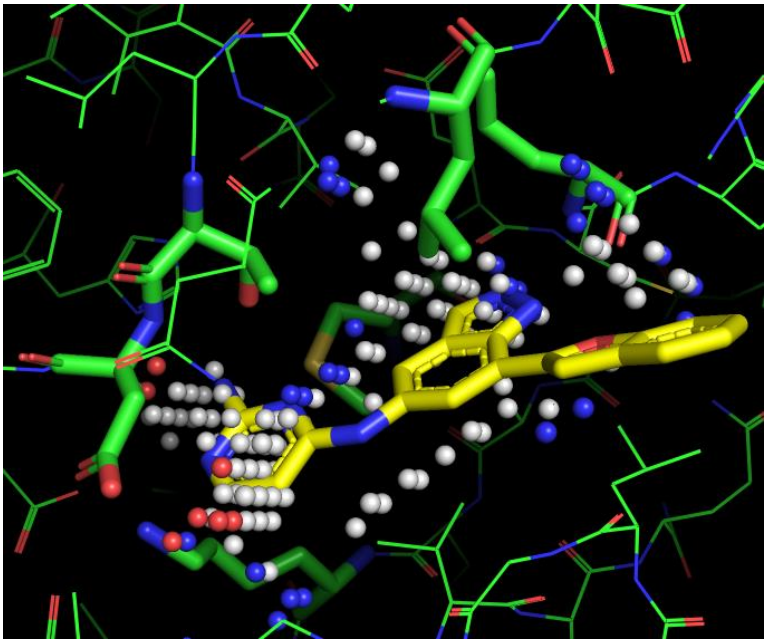


Figure 2. Snapshot of sitepoints in the binding pocket of MK2 in complex with N4-(7-(benzofuran-2-yl)-1H-indazol-5-yl)pyrimidine-2,4-diamine (PDB id: 3KC3) generated by LIDAEUS under default setting. Each sitepoint is coloured depending on built-in ligand-protein interaction (**Hydrophobic**: white; **HBD**: blue; **HBA**: red). Key amino acid residues are in bold. (**MK2**: MAP kinase-activated protein kinase 2; **PDB**: protein data bank; **HBD**: hydrogen bond donor; **HBA**: hydrogen bond acceptor)

Two SFs are integrated in LIDAEUS, that is, a force-field-based energy function and a knowledge-based function, the “pose interaction profile” (PIP). The former function is a linear combination of van der Waals and geometry-dependent hydrogen bonding interaction energies (**eq. 1** & **eq. 2**), which are inherited from the program PROBIS (Walkinshaw & Floersheim, 1990). The salt-bridge energy is also calculated through **eq. 2** and ionisable groups are identified by assigning fixed charges to them.

$$E(vdW) = \sum_g \left(\frac{A}{r_{gp}^{12}} - \frac{B}{r_{gp}^{6}}\right) \textbf{ (eq. 1)}$$

$$E(hb) = \sum_h \left(\frac{C}{r_{hp}^{12}} - \frac{D}{r_{hp}^{10}}\right) \cos\left(X_0 - X\right) \textbf{ (eq. 2)}$$

(**vdW**: van der Waals; hb: hydrogen bonding; **A, B, C and D** are coefficients taken from PROBIS; **X** is the ideal angle for maximum hydrogen bond energy)

In PIP, significant ligand-protein interactions can be converted into a hexadecimal code as a reference. Under default setting, these key features are exacted from the co-crystal structure of the target and a known ligand. New PIPs generated from the interaction between docked ligands and the protein, are subsequently compared to the reference PIP by calculating the Tanimoto coefficient. The final scores of LIDAEUS can be modified by the PIP scores for increasing the scores of poses matching the criteria. LIDAEUS, as a customizable virtual screening tool, is highly flexible to the screening of diverse targets, of which the applicability has been demonstrated by even recent research (Yang et al., 2007; Lim et al., 2011; Healy et al., 2015; Houston et al., 2015; Pearson et al., 2018)

AutoDock Vina (Trott & Olson, 2009) is a flexible-docking virtual screening program with faster screening speed, higher binding mode prediction accuracy and more precise affinity approximation than these of its predecessor AutoDock 4. The search algorithm adopted by Vina is titled “iterated local search global optimizer”, a stochastic optimization approach consisting of two successive steps. A mutation step, derived from genetic algorithm, generates variable ligand poses which are subsequently evaluated by a quasi-Newton method considering both the geometric arguments and corresponding energy of each ligand. The pose search of each ligand initiates from a random seed, the running steps of which varies in number depending on the extent of a ligand’s flexibility. Vina employs a grid-based SF, inspired by X-score (Wang et al., 2002), which tends to be a machine-learning-like algorithm rather than a simply physics-based energy calculation as this SF makes use of the principles from knowledge-based SFs and empirical SFs. It is parameterized by using experimental data from PDBbind, including binding affinity and structural information extracted from ligand conformations of available complexes. The advent of Vina has brought numerous success to molecular docking area and because of high adaptivity, Vina still outperforms numerous recent VS programs (Gaillard, 2018).

Protein flexibility

As introduced above, although early docking programs, like LIDAEUS, dock rigid ligand onto static protein structures, they still led to successful ligand discoveries, whereas subsequent flexible-docking programs, like Vina, taking into account the conformations of ligands, have earned greater success and broader recognition (Chen, 2015). It is notable that compared to the protein flexibility, the conformational space of ligands is quite limited, thus, multiple ligand conformers docking being straightforward. However, the thermal fluctuations of atoms in a protein receptor may result in a multitude of protein sub-states corresponding to different shapes of binding pocket (Amaro et al., 2018). Strategies including soft docking, rotamer libraries and optimization of partial side chains had been implemented to address this problem, which, essentially, all failed to produce significant backbone-level conformations of proteins (De Vivo et al., 2016). In this regard, molecular dynamics (MD) simulation has been employed to explore diverse states of the targets and had successfully unravelled a novel binding site of HIV-1 integrase (Hazuda et al., 2004). The common application of MD in VS is to facilitate ensemble docking by using the snapshots of protein provided by MD in a timescale to build a multiple conformations collection (Campbell et al., 2014). An apparent problem of this method is that MD may provide redundant conformations of a protein as a novel pocket is rarely observed at a 1-μs timescale (Śledź & Caflisch, 2018). To avoid this issue, it is advisable to shorten the timescale and conduct cluster analysis for snapshots ensemble, which may lead to inevitable computational time increasing (De Vivo et al., 2016).

Entropy calculation

When a ligand is docked into a binding pocket, it displaces some of the water molecules occupying the binding cavity accounting for a part of the binding free energy. Similarly, the entropy loss occurs when a flexible ligand is docked into the binding site. However, most force-field-based SFs ignore both the desolvation contribution and entropic effects, which are significant to

a more accurate ligand-binding free energy approximation (Drwal & Griffith, 2013). Indeed, due to the presence of implicit water molecules, direct calculation of desolvation contribution and entropic effects is less feasible, whereas an MD-based ligand binding free energy estimation approach, proposed by Kollman et al. (2000), has enjoyed high popularity among academia because of its applicability in solvent and entropic calculation, that is, the molecular mechanics energies combined with the Poisson–Boltzmann or generalized Born and surface area continuum solvation (MM/PBSA and MM/GBSA).

Binding process can be described by such reaction:

$$L + P \leftrightarrow PL$$

where L and P denote ligand and protein, respectively and PL is the ligand-protein complex.

With MM/PBSA, the binding free energy can be estimated from

$$\Delta G_{bind} = G_{PL} - G_P - G_L \text{ (eq. 3)}$$

where the free energy of each state, namely P, L or PL, can be decomposed into contributions of different interactions, solvation free energies and the entropic contribution (**eq. 4**).

$$G = E_{bnd} + E_{el} + E_{vdW} + G_{pol} + G_{np} - TS \text{ (**eq. 4**)}$$

($\boldsymbol{E_{bnd}}$: bonded energy from bond, angle and dihedral; $\boldsymbol{E_{el}}$: energy from electrostatic interaction; $\boldsymbol{E_{vdW}}$: energy from van der Waals interaction; $\boldsymbol{G_{pol}}, \boldsymbol{G_{np}}$: polar and non-polar contributions to solvation free energies; $\boldsymbol{T}$: the absolute temperature; $\boldsymbol{S}$: the entropy)

Notably, the electrostatic energy is calculated with a uniform dielectric constant. In addition, the polar contribution is typically obtained by the Poisson–Boltzmann (PB) equation in MM/PBSA or solved by using the generalized Born (GB) model of the MM/GBSA approach, whereas the non-polar energy is estimated by the solvent-accessible surface area (SASA) approach (Kollman et al., 2000). The entropic term is usually calculated by a

normal-mode analysis (Srinivasan et al., 1998) of protein conformational snapshots generated by MD simulation. In order to obtain a more precise affinity approximation, a standard protocol of MM/PBSA or MM/PBGA is to conduct independent simulation for the complex, the free ligand and unbounded receptor respectively and calculate each average free energy (**eq. 5**).

$$\Delta G_{bind} = \langle G_{PL} \rangle_{PL} - \langle G_P \rangle_P - \langle G_L \rangle_L \text{ (**eq. 5**)}$$

(**Brackets** denote the ensemble averages and the subscripts are the simulating targets)

A more computationally inexpensive approach is to only simulate the complex and calculating the average free energies of free ligand and receptor generated by simply removing related atoms from the complex (**eq. 6)** (Genheden & Ryde, 2015), where the conformational changes of ligand and receptor are neglected.

$$\Delta G_{bind} = \langle G_{PL} - G_P - G_L \rangle_{PL} \text{ (**eq. 6**)}$$

Despite the high popularity, MM/PBSA and MM/GBSA may suffer from poor precision because of oversimplified conformational and solvation entropic estimations, ignorance of ions and key water molecules in binding pocket and the use of a unified dielectric constant for electrostatic energy calculation. By contrast, higher precision requires larger computational resources, including computing time and memory, rendering MM/PBSA and MM/GBSA less feasible for being employed in high throughput virtual screening. Thus, further improvements and reasonable trade-off of precision and computing efficiency should be made to these methods and it might be more suitable employing MM/PBSA and MM/GBSA in post-processing (Genheden & Ryde, 2015; Wang et al., 2019). Therefore, a more robust and less computationally expensive energy estimation in high throughput virtual screening should be developed.

Performance evaluation of SBVS methods

As SBVS has been intensively employed in early-stage drug discovery, the reliability of virtual screening tools has become increasingly significant. Before evaluating the performance of a VS tool, it is worth re-addressing the objective of conducting a VS research, which is to enrich a sub-set of compounds as potential actives from a large chemical pool in order to reduce the number of inactive molecules used in HTS.

With this regard, a variety of data sets composing of actives with experimentally confirmed affinity and presumed or confirmed inactive compounds, also known as decoys, against specific receptor targets, have been used in VS tools' performance evaluation as the "chemical pool", with this compound set known as a "Benchmarking data set" (Lindh et al., 2015) and methods evaluating the enriching power or ranking power are called "performance metrics".

Benchmarking data set

There are two main problems in benchmarking data, which lead to benchmarking biases, that is, "artificial enrichment" and "analogue bias", both of which can lead to overoptimistic results (Xia et al., 2015). Due to overlarge physicochemical differences between actives and decoys in benchmarking data sets, SBVS methods with poor SFs may still succeed in discriminating active and inactive compounds, which is so-called "artificial enrichment", whereas "analogue bias", resulting from highly similar actives occurring in data sets, may lead to partial enrichment in VS results.

Here, we will introduce approaches used by DEKOIS 2.0, benchmarking data sets employed in our project, as illustrations on avoiding benchmarking biases.

DEKOIS 2.0 (Bauer et al., 2013) is a benchmarking library containing diverse ligands against 80 protein targets varying from ligases to kinases, where the bioactivity data of ligands are extracted from BindingDB (Liu et al.,

2007). To avoid the “analogue bias”, variable active ligands against the same target are firstly clustered into 40 groups by comparing FCFP_6 Tanimoto similarities. The most potent ligands from each cluster are subsequently grouped as the active compounds data set against this target. DOE score (Vogel et al., 2011), calculating molecular distances by using multidimensional physicochemical properties, is employed to evaluate the extent of difference between active compounds and decoys. Decoys with lower DOE scores (< 0.1) indicate closer physicochemical space between actives and decoys, and less possible occurrence of “artificial enrichment”. Besides “analogue bias” and “artificial enrichment”, latent active decoys should be avoided in benchmark data sets (Scior et al., 2012). In DEKOIS 2.0, decoys with least structural similarity to actives are selected in order to eliminate latent bioactive compounds.

Metrics

Normally, the performance of a SBVS tool should be evaluated under two criteria, namely, successful enrichment of actives and accurate pose of ligands (Lagarde et al., 2015). Given broad acknowledgement that the docking algorithms are able to produce relatively accurate ligand binding modes (Matter & Sotriffer, 2001; Cheng et al., 2012; Ferreira et al., 2015), mainstream VS performance metrics attempt to quantify the ability of a VS tool to rank actives before inactive compound. With this regard, VS tools have been treated as binary classifiers and molecules, which have already tagged with “active” and “inactive” in the benchmarking data sets, are ranked by their scores, generated by VS programs, in the results.

The ideal VS program should always rank active compounds before the decoys and an apparent score-based threshold should be well presented. However, most score distributions may perform like **Figure 3a**, where partial actives might be observed at the top places, yet considerable amount of actives sharing similar scores with the decoys. Thereby, the performance of a VS program varies at different thresholds. With this regard, receiver operating characteristic (ROC) curve, a metric widely applied in the assessment of binary classifiers, has been intensively employed in VS overall performance evaluation (Triballeau et al., 2005; Fawcett, 2006). In VS assessment, the ROC curve is essentially a cumulative curve reflecting the occurrence of actives at different threshold **(Figure 3, c**). Considering a quantitative assessment, the area under the ROC curve (AUC) is calculated and interpreted as the probabilities that an active has a higher score than a decoy has when they are both randomly selected (Triballeau et al., 2005). Although the AUC calculation of ROC curves has gained great popularity in VS assessment, given that it ignores specific consideration on VS's objective, its applicability is restricted in overall evaluations (Truchon & Bayly, 2007).

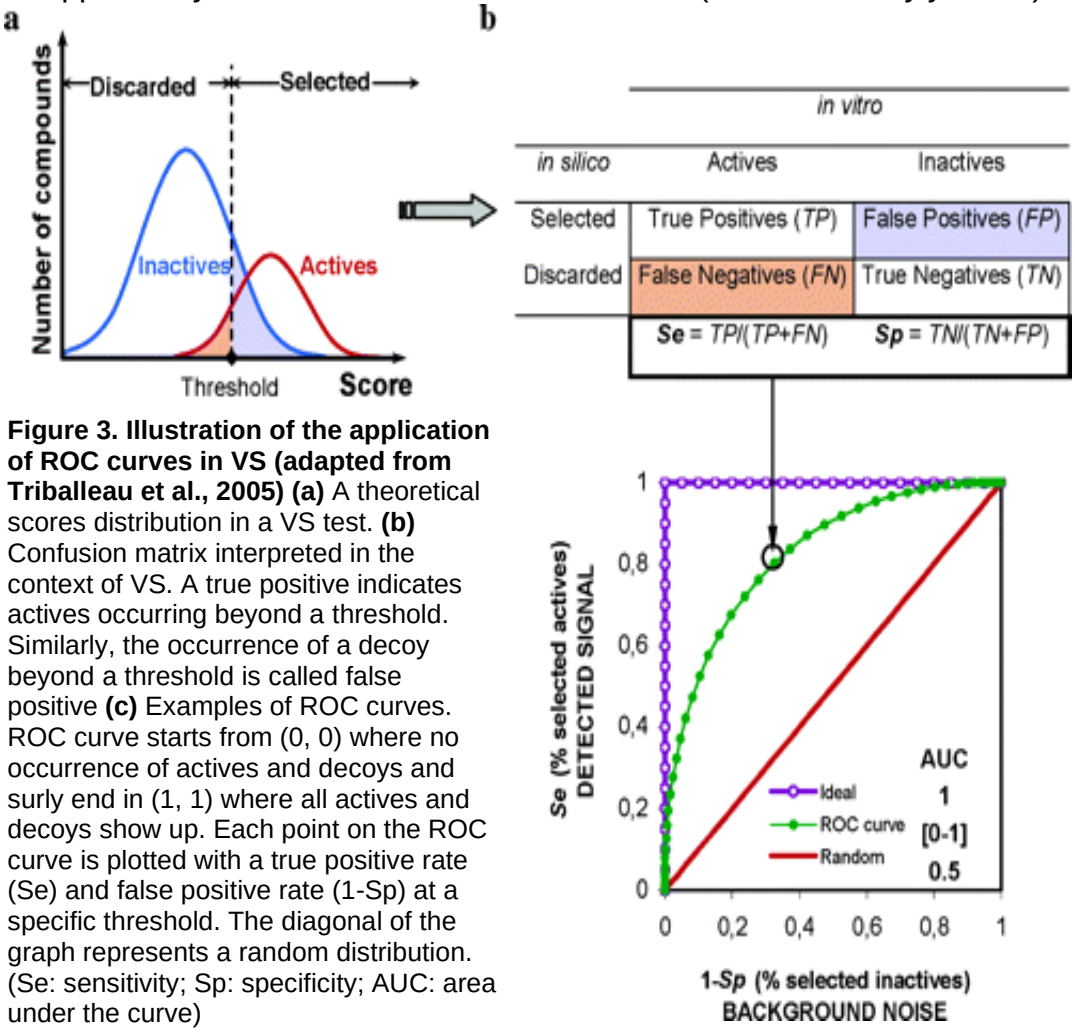


Figure 3. Illustration of the application of ROC curves in VS (adapted from Triballeau et al., 2005) (a) A theoretical scores distribution in a VS test. **(b)** Confusion matrix interpreted in the context of VS. A true positive indicates actives occurring beyond a threshold. Similarly, the occurrence of a decoy beyond a threshold is called false positive **(c)** Examples of ROC curves. ROC curve starts from (0, 0) where no occurrence of actives and decoys and surly end in (1, 1) where all actives and decoys show up. Each point on the ROC curve is plotted with a true positive rate (Se) and false positive rate (1-Sp) at a specific threshold. The diagonal of the graph represents a random distribution. (Se: sensitivity; Sp: specificity; AUC: area under the curve)

"Early recognition" problem consideration

Due to the limitation of resource, under most circumstances, only a limited portion of the VS results will be further experimentally tested (Ain et al., 2015), which introduces so-called "early recognition" problem into SBVS performance evaluation. It has been intensively documented that the observation on the shape of ROC curve can tell whether a VS tool addresses "early recognition" problem, yet, this method fails to effectively compare VS programs (Walter, 2005; Truchon & Bayly, 2007; Kirchmair et al., 2008; Nicholls, 2008). Thereby, metrics addressing this problem have been explored and employed.

Enrichment factors (EF) is a metric specializing on "early recognition" problem, introduced by Halgren et al. (2004). EF only focus on the number of actives in the top $x\%$ of the compounds in the ranking lists (**eq. 7**) and this top $x\%$ defines the earliness of the "early recognition". However, it is also criticized for the maximum value highly dependent on the ratio of actives and total compounds and the ignorance of the scores possibly leading to VS tools with different ranking performance sharing the same EF (Kirchmair et al., 2008).

$$EF = \frac{N_{actives\ in\ top\ x\%} / N_{compounds\ \ in\ top\ x\%}}{N_{total\ actives} / N_{total\ compounds}} \textbf{ (eq. 7)}$$

(**N** denotes the number)

Robust initial enhancement (RIE) (Sheridan et al., 2001) was created to provide a metric with avoidance of large value variations resulting from little amount of investigated actives (Kirchmair et al., 2008). Thereby, ranked actives are assigned with different weights according to the ranking and the number of investigated actives α. All the weights are subsequently summated as S (**eq. 8**).

$$S = \sum_{i=1}^{actives} e^{-\left(\frac{rank(i)}{\alpha}\right)} \textbf{ (eq. 8)}$$

Final RIE score is calculated in **eq. 9**

$$RIE = \frac{S}{\langle S \rangle} \text{ (eq. 9)}$$

where the mean sum $\langle \boldsymbol{S} \rangle$ is derived from random distribution ranking. The RIE score indicates the extent that VS methods are better/worse than a random distribution. Since earlier ranked actives will be assigned larger weights, VS methods providing earlier enrichment will have higher RIE score than those failing to rank most actives at the beginning. However, RIE still suffered from the same disadvantages as EF, that is, the dependence on actives ratio and the ignorance of the entity performance (Jain, 2008).

Boltzmann-enhanced discrimination of receiver operating characteristic (BEDROC) was created by Truchon & Bayly (2007) with the aim of addressing "early recognition" problem but still considering overall performance. The BEDROC metric is a generalized-form of AUC, including a decreasing exponential weighting function similar to RIE (**eq. 10**). The idea of BEDROC is to assign large weights to a specific portion of ranking when the score is calculated, which is controlled by a tuneable parameter α.

$$BEDROC = RIE \times \frac{R_\alpha \sinh\frac{\alpha}{2}}{\cosh\left(\frac{\alpha}{2}\right) - \cosh\left(\frac{\alpha}{2} - \alpha R_\alpha\right)} + \frac{1}{1 - e^{\alpha(1-R_\alpha)}}$$

$$\approx \frac{RIE}{\alpha} + \frac{1}{1-e^{\alpha}}, if\ \alpha R_\alpha \ll 1\ and\ \alpha \neq 0 \text{ (eq. 10)}$$

In order to compare different VS methods with different actives distributions, αR_α should be very much less than one (Truchon & Bayly, 2007). Truchon & Bayly (2007) also suggested a value of 20 for α, indicating the top 8% of total compounds contributing to 80% of the total BEDROC score. However, for a meaningful BEDROC value, the ratio of actives should be much less than 5% according to $\alpha R_\alpha \ll 1$, which might not be met by many benchmark data sets. Correspondingly, when the ratio of actives is fixed, the earliness of interest is limited. Besides, although the value of

BEDROC is different from the RIE value, it is proved that they are in a linear correlation (Zhao et al., 2009).

To address “early recognition” problem, numerous metrics have been borrowed from other fields or reformulated. It is clear that there is no consensus metric for this problem so far. In the assessment of VS tools’ performance, a more modest result could be generated by analysing with different metrics.

Target considerations after VS performance evaluation

Investigations have revealed that target-dependent performance prevails among the results of different docking programs’ assessment (Rognan, 2013). It is hard for search algorithms and SFs to exhibit consistent performance across different classes of targets. Therefore, identifying possible target families, where a VS program can offer its best performance, may be significant for obtaining accurate results in blind screening. However, it is notable to stress that herein target comparison aims at finding geometrically similar binding sites rather than the whole target structures.

Comparison of binding sites

Besides the above benefits for a VS tool, comparing binding sites between proteins with no obvious global structural similarities can also provide possibilities for small molecules being multi-targeting actives and expand the compound classes and scaffolds for the targets (Siragusa et al., 2016). Generally, binding site comparison methods can be categorized according to different descriptors applied, such as fingerprints, graphs, grids and other alternatives, which can impose significant impacts on both the outcomes and the computational expense. (Ehrt et al., 2018). A precise result requires employing MD to account for the targets’ flexibility and explore implicit binding sites for *apo* structures of proteins (Ehrt et al., 2016). Here, we attempt to utilize a robust and fast method addressing such binding site comparison problem.

Summary

An ideal VS program is, undoubtedly, the one being able to predict relatively correct affinity of ligands, thereby lowing the number of inactive compounds being tested in HTS, but, in our project, the expectation for an ideal VS tool is just to distinguish active compounds and decoys. With this regard, the objective of our project is to evaluate the VS performance of LIDAEUS in a benchmarking system, where Vina is deemed as the reference *in silico* screening program. Additionally, we attempted to make improvements to LIDAEUS's performance in such controlled system and to judge whether the metrics employed in the evaluation actually address the real-world problem.

Methods & Materials

In our project, all computational experiments were conducted on machines with x86 architecture processors and 64-bit operating systems, that is, Scientific Linux with GNU bash, version 7.4 (Nitrogen), and Windows 10.

Preparation for benchmarking data

The process of preparing input receptor structures is outlined as a workflow in **Figure 4**. All protein structures in our project were obtained from RCSB PDB (Burley et al., 2018). The PDB entries provided by DEKOIS 2.0 (Bauer et al., 2013) were used except the structure for NAD-dependent deacetylase sirtuin 2 (**SIRT2**), of which an unbound (*apo*) structure (PDB id: 1J8F) was offered. Therefore, appropriate structure of SIRT2 was derived by carrying out a protein BLAST search on RCSB PDB using the protein sequence of the *apo* structure. BLAST E-value and sequence identity cut-off were set to 10.0 and 90% respectively. Low complexity regions were not masked and only ligand-bound (*holo*) structures were shown. Crystallographic parameters, namely, the R-values and structural resolution, and the activity of ligands were considered in the selection of suitable alternative.

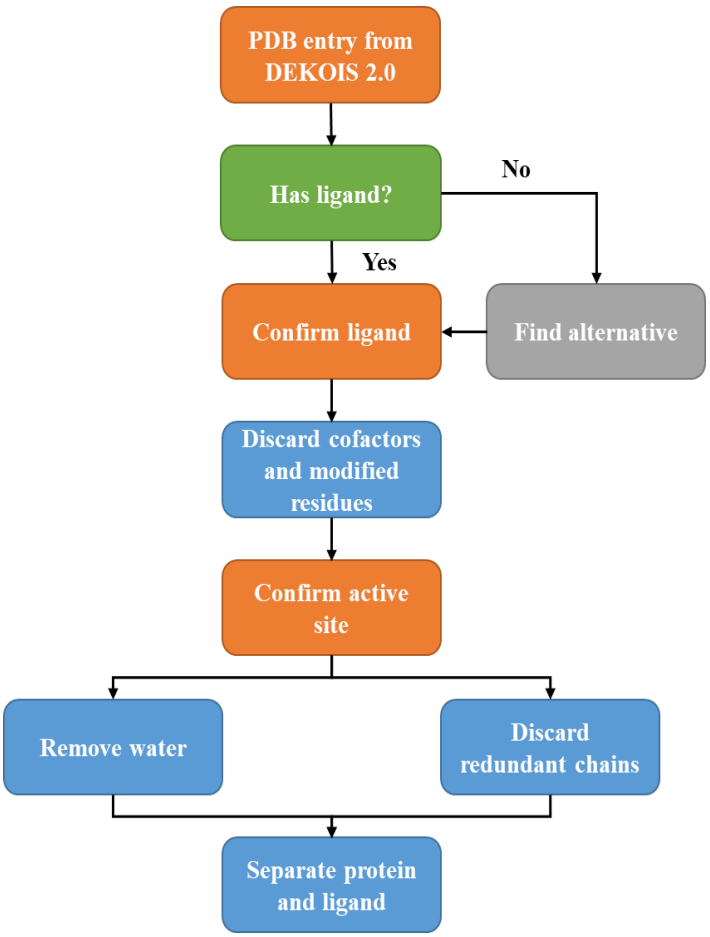


Figure 4. Workflow of receptor structure preparation.

Co-crystallized ligand and the location of active site of each protein were confirmed by manually reviewing related literatures. All receptor

structures were processed in PyMOL version 2.3.2 (Schrödinger, LLC., 2019) with discarding redundant and ligand-unbound chains and the removal of all cofactors, modified residues and water molecules. Ligand and protein were separated and saved ad PDB files respectively. For ligand crystals with multiple orientations, one arbitrary structure was kept. For Vina, input structures were further protonated at pH 7.4 by employing a Python script (pdb2pqr.py version 2.1.1) and converted into PDBQT format using another Python script (prepare_receptor4.py) provided by the MGLTools package (version 1.5.6) (Morris et al., 2009). For each target, there are 40 actives and 1200 decoys in the benchmarking data set, excluding severe acute respiratory syndrome-associated coronavirus (**SARS-HCoV**) data set which has only 39 actives and 1170 decoys. Actives and decoys were combined into one SDF file where the top 40 compounds (39 in the case of SARS-HCoV) were actives and the other molecules were decoys. Hereby, the ratios of actives among targets are all 0.0323.

Rigid docking of LIDAEUS

For every target, the sitepoints and energy maps were generated by using the co-crystallized ligand for each receptor with default parameters (**Table 1**). Docking was performed with default settings where the resolution of fitting tolerance (resol) was set to 0.02. In order to increase the chance of obtaining a docked version of each actives with their best energy scores, the maximum number of results was set to 50,000.

Parameter	Meaning	Value
Pad	the depth of sitepoints	1.5
espSPC	the energy cut-off of special carbon	-1.15
espC	the energy cut-off of normal carbon	-1.0
espHBA	the energy cut-off of hydrogen bond acceptor	-4.8
espHBD	the energy cut-off of hydrogen bond donor	-4.8
Spopt	the number of sitepoints	170

Table 1. Default parameters of the sitepoint generation in LIDAEUS.

Flexible docking of LIDAEUS

In order to explore its capability of flexible docking, ligand conformers were generated by OpenBabel version 2.4.1 (O'Boyle et al., 2011) using a genetic algorithm with the generation of nine conformers optimized for energy diversity for each ligand. To compare LIDAEUS with Vina in regards to flexible docking performance, the result from Vina of each target was used as the input ligand data in LIDAEUS.

Flexible docking of Vina

In Vina, the docking algorithm explores in a defined search box, which, in our project, was located by using the centre of gravity (COG) of the corresponding sitepoints generated in LIDAEUS. The size of the search box was calculated by the program boxmol with a tuneable parameter padding (Author personal communication, 2019). Given that Vina program only docks a single compound per run, a script was used to run the program across all compounds for each DEKIOS 2.0 data set (Author personal communication, 2019). To investigate Vina's docking performance in regards to the sitepoints dimensional variants, six groups of sitepoints with different COGs, derived by setting different energy cut-offs in LIDAEUS and setting padding to 2 in boxmol, were adopted in a series of screenings against B-cell lymphoma 2 (**BCL2**). In the assessment of the influence of the search box's volume on Vina's docking performance, all the COGs of these search boxes were derived from the default sitepoints, whereas, the dimensions of these search boxes were incremented or decremented by 4 Å.

Evaluations of VS performance

All the docking results were evaluated by calculating AUC-ROC, BEDROC and EF respectively. The calculation of AUC-ROC was implemented in Python according to Fawcett (2006) (Code in **Appendix 6**). Modules from RDKit (Open-source cheminformatics, 2019) were used to calculate BEDROC and EF. Due to multiple orientations or conformers of a single compound was generated by using LIDAEUS or Vina, only one

orientation or conformer with the highest affinity or energy score was shown in the final ranking list. Compounds that did not appear in the results were manually added to the end of the ranking list with extremely low scores and actives were added first. In LIDAEUS's results, compounds named with "BDB" were considered as actives. As part of the processing for Vina, the compound names were lost, however, each result file is related to the original compound by a number indicating the order in the input file of the script, so in Vina's results, compounds named with number less or equal to 40 (39 in the case of SARS-HCoV) represent active ligands. The performance of VS tools were classified into four groups according to the AUC-ROC derived from the screening of each target, namely, "**distinction**", "**merit**", "**pass**" and "**fail**", corresponding to AUC-ROC higher or equal to 0.70, between 0.69 and 0.60, over 0.50 but lower than 0.60 and lower than 0.50.

Three different α were adopted in BEDROC calculations with different aims. α of 1, was calculated from $\alpha R_\alpha \ll 1$ where the ratio of actives is 0.0323, following the suggestion from Truchon & Bayly (2007). To focus on the "early recognition" problem, α of 160 and 20 were chosen to evaluate the scoring power of VS tools within the top 1% and 8% of the compounds in the results, whereas, the top 1% to the top 18% of which was measured by EF. To judge the adaptivity of these three metrics, Microsoft Excel (Microsoft Corporation, 2016) was employed to visualize the results from different metrics and calculate the Pearson correlation coefficients of them.

Attempts to improve LIDAEUS's performance

Targets, where the performance of LIDAEUS fell into "**fail**" group, were selected for re-docking for better results with different settings.

Result-lead optimization experiment was carried out by incrementing or decrementing the values of the parameters in the generation of sitepoints, except the spopt, according to the increase or decrease of the AUC-ROC obtained from last change. Given an arbitrary change of *k* made to a parameter *w*, if the resulting AUC-ROC increases, the subsequent

change of *w* will still be *k* until the AUC-ROC decreases and the next change will be –*k*. When the AUC-ROC decreases twice successively, the modification will be made to another parameter. The order of changing parameters is shown in

$$espHBA\ and\ espHBD\ >\ espC\ >\ espSPC\ >\ pad \quad \textbf{(eq. 11)}$$

where *espHBA* and *espHBD* were deemed as a unity. Each time, only one parameter was modified and the change was always ± 0.1. Exception was made to a series of dockings against retinoid X receptor (**RXR**), where the changes were unbound.

Sitepoints variants on VS performance of LDIAEUS

Multiple screenings against RXR were performed while varying the fitting tolerance (tol) and the number of sitepoints (spopt). For each parameter, systematic changes were made to investigate the parameter variant influence on LIDAEUS's performance. The influence of the point types and the point distribution on the quality of screening results was investigated by replacing the type of points in the sitepoints generated by optimized parameters with the type of their closest points in sitepoints generated by default parameters. The resulting sitepoints were used in subsequent docking. Sitepoints generated with default parameters in the docking against RXR and Erb-B2 Receptor Tyrosine Kinase 2 (**ERBB2**) were respectively re-shaped by arbitrarily changing the coordinate of each sitepoint, which were both repeated for 1,000 times to explore the influence of sitepoints' shape on docking performance. Further shape changes were made by randomly selecting two sets of sitepoints derived from the previous 1,000 sets and combining them together to generate a new set of sitepoints, which was conducted in the case of ERBB2 and RXR for 100 times respectively.

Re-scoring

The separated ligands in receptor preparation process was used as the reference ligands in PIP re-scoring to explore key contacts of ligands and the proteins, which is conducted by a module of LIDAEUS. Four different Tanimoto coefficients were calculated based on the docked contracts and the reference contacts. Results ranked by different PIP scores were evaluated by calculating AUC-ROC. Among targets, because interactions with metals were not record in LIDAEUS, PIP re-scoring was not applicable to SARS-HCoV where zinc is the only contacting atom.

In the assessment of re-calculating the free energies, different weights were assigned to different components of each ligand's energy, namely, the van der Waals, hydrogen bond donors' energy and hydrogen bond acceptors' energy. Each weight ranged from 0 to 20 with an increment of 1, resulting in 9261 outcomes for each target. When considering PIP scores, the energy calculation was performed in

$$Free\ Energy = PIP_x \times (a \times vdW + b \times HBD + c \times HBA) \quad \textbf{(eq. 12)}$$

where *x* represents the kind of PIP score and *a*, *b* and *c* are the different weights ranging from 0 to 20 with increment of 1. Here, in order to save computational time, weights combinations with the same ratio of *a, b* or *c* were calculated once, where 30,056 results were generated for each target. AUC-ROC was used to evaluate each outcome (Code in **Appendix 12**).

To calculate the entropy penalty for each binding free energy, a simple energy re-scoring method was carried out by counting the loss of water molecules and calculating entropic loss of contacting amino acids based on the annotated values from Doig and Sternberg (1995), using a Python program (Code outlined in **Appendix 8**). The hydrogen bond donor atoms and the hydrogen bond acceptor atoms of both amino acid residues and ligands were assigned with associated amount of water molecules according to the setting in **Appendix 8**. Each water molecule accounts for 3 kcal per molar at 300K (Dunitz, 1994). When the distance between an amino acid

residue atom and a ligand atom was less than 3.5 Å, one water molecule was assumedly lost and the entropic loss of this residue was counted. Ultimately, the enthalpy, water loss entropy and side chain conformational entropy were assigned systematically changed weights aiming to find out the best parameters for each target (**eq. 13**). Similarly, further calculation was considering four PIP scores (eq. 14). (Codes outlined in **Appendix 12**).

$$Free\ Energy = a \times (vdW + HBD + HBA) - (b \times T\Delta S_{water\ loss} + c \times T\Delta S_{side\ chain})$$ (**eq. 13**)

$$Free\ Energy = PIP_x \times weighted(Enthalpy + Entropy)$$ (**eq. 14**)

Large-scale screening

A large-scale screening against Estrogen receptor beta (**ER-beta**) was performed by LIDAEUS, using Maybridge Screening Collection (Thermo Fisher Scientific), which contains 54,028 pre-synthesized compounds, as the decoy data set, and active ligands targeting on ER-beta from DEKOIS 2.0 data set. RDkit was employed to compare the Morgan fingerprints extracted from decoys and actives to eliminate decoy molecules that are similar to the active compounds in physicochemical features. Vina was subsequently used to screen the first 1,000 molecules in LIDAEUS's result. The AUC-ROC and the number of active in the range of interest of each screening outcome were calculated respectively. Such experiment is intended to simulate the real-world VS project that Houston et al. (2015) did.

Binding site extraction

In order to extract relatively complete binding sties, the available crystallographic poses of ligands in the DEKOIS 2.0 actives set were obtained in the workflow of **Figure 5 (left)** using a Python script (Code outlined in **Appendix 9**). The obtained co-crystal structures were subsequently processed in the workflow of **Figure 4**. Method “align” in PyMOL was used to align these processed co-crystals onto the receptor structure adopted in docking, which agglomerated all the active ligands. Residues of the docking protein structures, including cofactors, around this agglomeration within 3.5 Å were deemed as contacting residues and the components of the binding site (**Figure 5, right**). Above operations were carried out by using a Python PyMOL script (**Appendix 10**).

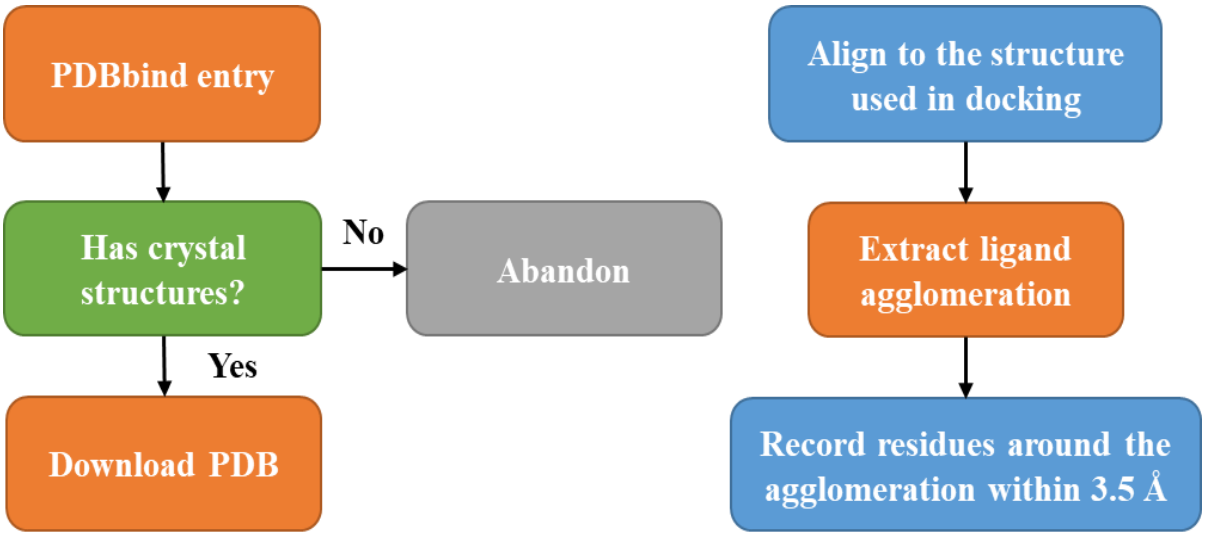


Figure 5. Process of binding site extraction. Left: collecting crystals with known ligands against the same target. **Right**: finding contacting amino acid residues to build a binding site.

Binding site classification

Two classification strategies were conducted by employing component-based method and geometry-based method. In component-based method (**Appendix 2**), amino acids were initially categorized into 5 groups depending on the chemical characteristics of their side chains, that is, hydrophobic aliphatic, hydrophobic aromatic, polar uncharged, positively charged and negatively charged. The other contacting residues were grouped into metals or other cofactors. Binding sites were then classified into

hydrophobic or polar binding sites based on the properties of the quantitatively principle components. According to whether an amino acid residue carries atoms that could be hydrogen acceptors (HBA) or hydrogen bond donors (HBD), amino acids were further classified into three types, namely, hydrophobic, HBA, and HBD. Correspondingly, the type that dominated binding sites in numbers decided the classes of the binding sites. In order to obtain quantitative results, the hydrophobicity of each binding site was calculated by summing up the hydrophobicity of each amino acid residue assessed by Fauchere and Pliska (1983).

Binding sites geometric comparisons were implemented in a modified ultra-fast shape recognition Python program (Code outlined in **Appendix 11**) according to the algorithm introduced by Ballester et al. (2009) and further improved by Shave et al. (2015). Each binding site was treated as a whole molecule, of which the centroid (ctd), the closest atom to the ctd (cst), the farthest atom from the ctd (fct) and the farthest atom from the ctd (ftf) were determined. The distribution of Euclidean distances from other atoms to the above four atoms was measured successively and the related mean, variance and the skew of each atomic distance distribution were calculated respectively. Thereby, a binding site was depicted with twelve descriptors, which were used to calculate the Euclidean distances between binding sites. In the improved published version, the number of descriptors were increased to 48, whereas, in the implementation, we used 36 descriptors derived by calculating coefficients on sets of hydrophobic atoms, hydrogen bond acceptors and hydrogen bond donors. All similarity matrices were visualized by using Microsoft Excel (Microsoft Corporation, 2016). Targets were subsequently clustered by using each classification methods, the visualisation and average-distance hierarchical clustering operation of which was conducted by a Python module Plotly (Plotly Technologies Inc., 2015).

Results

Preparation for benchmarking data

Details of benchmarking data preparation were exhibited in **Appendix 3**. In the process of preparing target crystal structure, active sites with cofactors were found in 14 targets (**Table 2**). Due to the presence of cofactors leading to the malfunction of sitepoint generation system, all cofactors including modified residues were discarded. The sitepoint generation system failed to handle ligands with multiple orientations (**Table 3**) leading to disordered and irrational sitepoints (**Figure 5**), due to which, one of each ligand's orientations were arbitrarily selected for sitepoint generation. Similarly, ligands containing atoms that are not included in LIDAEUS's force field tables also resulted in such randomly distributed site points. Besides, the output Structure Data Format (SDF) file of LIDAEUS may lose the integrity of the format under uncertain situation, leading to the disappearance of some compounds in the molecular graphics software.

Targets	PDB id	Cofactor in active site
11betaHSD1	3TFQ	NAP
ALR2	1AH3	NAP
CATL	3BC3	CSD (Modified residue)
CYP2A6	1Z11	HEM
INHA	1P44	NAD
TS	1I00	UMP
ACE	1UZE	Zn
ACE2	1R4L	Zn
ADAM17	3EWJ	Zn
HDAC2	3MAX	Zn
HDAC8	3NU3	Zn
MMP2	1HOV	Zn
QPCT	2AFX	Zn
SARS-HCoV	2Z94	Zn

Table 2. Target structures with cofactor in active sties. NAP: Nicotinamide adenine dinucleotide phosphate; **CSD**: Cysteine sulfinic acid; **HEM**: Protoporphyrin IX containing Fe; **NAD**: Nicotinamide adenine dinucleotide; **UMP**: 2′-Deoxyuridine 5′-monophosphate.

Targets	PDB id	Ligand
17betaHSD1	3KLM	DHT
JNK1	3ELJ	GS7
TK	1W4R	TTP

Table 3. Structures with multiple-orientation ligands. DHT: 5-alpha-Dihydrotestosterone; **GS7**: 2-fluoro-6-{[2-({2-methoxy-4-[(methylsulfonyl)methyl]phenyl}amino)-7H-pyrrolo[2,3-d]pyrimidin-4-yl]amino}benzamide; **TTP**: Thymidine-5'-triphosphate.

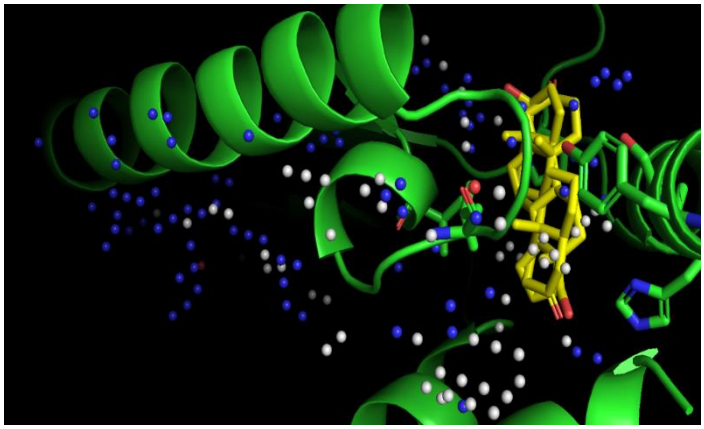


Figure 5. Snapshot of disordered sitepoints generated by using ligand with two orientations in the active site of 17betaHSD1 under LIDAEUS's default setting. Each sitepoint is coloured depending on built-in ligand-protein interaction. (**Hydrophobic**: white; **HBD**: blue; **HBA**: red). Key amino acid residues are in bold. (HBD: hydrogen bond donor; HBA: hydrogen bond acceptor)

Above deficiencies of the sitepoint generation system resulted from the lack of information stored in LIDAEUS and software errors. Undefined molecules or atoms in the target structures can lead to the termination of LIDAEUS before conducting docking. The file conversion program of LIDAEUS suite can produce false files dealing with unusual ligands to LIDAEUS, including the ones of multiple orientations or unidentified atoms, which resulted in the generation of chaotic sitepoints.

During manual confirmation of active sites, three targets with high-flexibility binding sites were confirmed by literatures (**Table 4**). Since the flexibility of target structures was not taken into account in our project, some unsatisfactory performance of both LIDAEUS and Vina might be attributed to such issue.

Targets	PDB id	Flexibility
HIV1RT	1S6P	Confirmed
HSP90	1UY6	Confirmed
VEGFR2	3C7Q	Confirmed

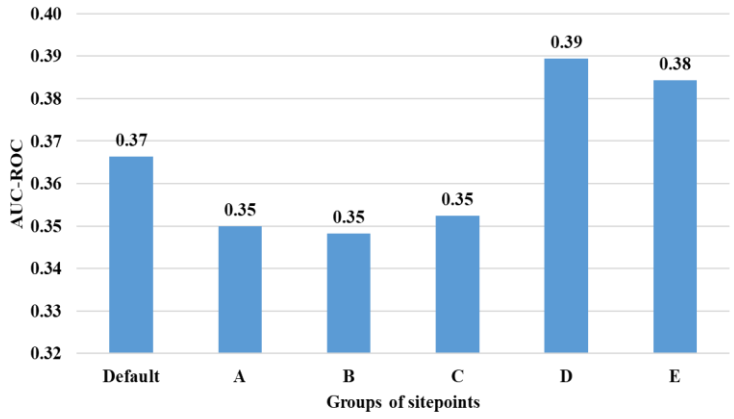
Table 4. Targets with high-flexibility binding sites. The flexible binding sites of HIV1RT, HSP90 and VEGFR2 were confirmed by Ivetac and Andrew (2011), Biamonte et al. (2009) and Sarabipour et al. (2016), respectively.

Assessing the impact of parameter variation on Vina's virtual screening performance

Multiple flexible docking experiments against B-cell lymphoma 2 (**BCL2**) were performed with Vina by using search boxes derived from various approaches. Each screening result was evaluated by using AUC-ROC metric. Ranking lists that only contain partial input compounds were manually complemented by adding missing compounds with the same energy score to the end of the list. Actives were added first in the manual operation. **Figure 6** reveals that the VS performance of Vina can be influenced by the dimensional variance of the search box. In **Figure 6 (B)**, although the centre of gravity (COG) of each set of sitepoints is similar, due to the shape differences, the volume of resulting search box varies, which led to the unstable performance of Vina shown in **Figure 6 (A)**. Notwithstanding that larger search space might eliminate the variance introduced by different

search box location, such assumption cannot be proven in **Figure 6 (A & B)**, where a search box with volume of 3299.0992 Å^3 resulted in a AUC-ROC value of 0.37 while a higher value of AUC-ROC (0.39) was derived from a smaller search box.

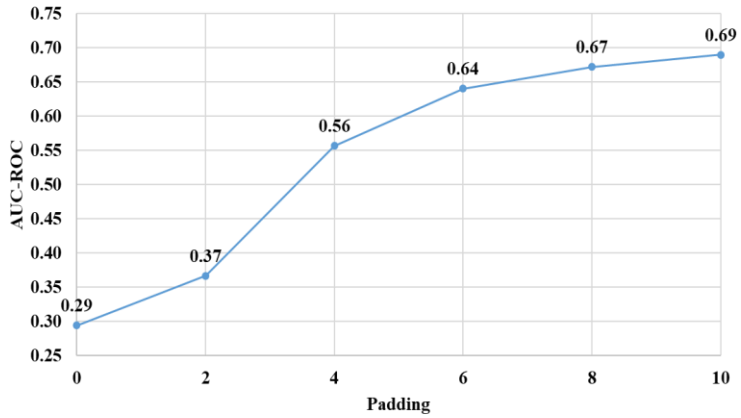
A



B

Groups	COG X (Å)	COG Y (Å)	COG Z (Å)	Volume (Å^3)
Default	38.509	26.422	-13.182	3229.0992
A	38.251	26.539	-12.712	3044.3495
B	38.427	26.334	-12.550	3122.3055
C	38.376	26.280	-12.604	3058.5280
D	38.311	26.235	-12.649	2975.1458
E	38.403	26.302	-12.582	2966.6736

C



D

Padding	X size (Å)	Y size (Å)	Z size (Å)	Volume (Å^3)
0	10.627	11.336	9.919	1194.9188
2	14.627	15.336	13.919	3122.3055
4	18.627	19.336	17.919	6453.9162
6	22.627	23.336	21.919	11573.7509
8	26.627	27.336	25.919	18865.8095
10	30.627	31.336	29.919	28714.0922

Figure 6. The impact of the search space dimensional variance on the docking performance of Vina. A: groups of sitepoints derived by using different parameters was used to generate the search box for Vina with padding being set to 2. The centre of gravity of each set of sitepoints and the associated volume are shown in **B**. Due to the location and volume variance co-varied in such experiment, the dominant factor affecting Vina's performance cannot be observed. Given the stochastic searching algorithm adopted by Vina, the AUC-ROC variation might not exclusively be caused by above dimensional variance. **C**: every 2 padding represents a 4 Å increment on each dimension of the search box modified on a box with 10.627×11.336×9.919 Å dimensions derived from LIDAEUS's default sitepoints generation. Resulting size of XYZ and related volume of each search box are shown in **D**. When the location of the search box was fixed, the extension of the search space can improve Vina's VS performance. However, the improvement made by the dimension increment was decreasing as the volume of the search box reached a certain scale. Here, after the volume exceeding 11573.7509 Å^3, the AUC-ROC gap between experiment groups dropped to 0.03 and 0.02, whereas, before this volume, the gaps were 0.08 and 0.19. **Note**: all the AUC-ROC values were round to 2 decimal places.

The correlation of the search box size and Vina's VS performance was investigated by systematically changing the XYZ dimensions of a search box

with a fixed COG derived from the default sitepoints generation in LIDAEUS. **Figure 6 (C & D)** shows that it was the over-narrow search space limiting the VS performance of Vina rather than a larger space making improvements to Vina's VS performance as a plateau of AUC-ROC can be observed when the box size is expanded to an extent. Thereby, all subsequent docking experiments performed with Vina used padding equal to 10 and sitepoints in LIDAEUS generated with default parameters to generate the search box.

Justification of the selection of VS program evaluation Metrics

AUC-ROC, BEDROC (α equal to 1, 20 and 160.9) and EF (1% and 8%) were calculated from LIDAEUS's rigid-docking result and Vina's flexible-docking result, the correlations of which were investigated by calculating their Pearson coefficients for selecting suitable metrics in the subsequent evaluation of VS tools (**Figure 7**). In order to obtain more meaningful BEDROC values, the value of α was first set to 1 rendering $\alpha R_\alpha \ll 1$, which means that 80% of the total BEDROC scores come from the top 80% of the ranking. Pearson coefficients calculation shows that, in the results of both LIDAEUS and Vina, BEDROC with this α is highly linearly correlated with the AUC-ROC (**Figure 7, A & G**), indicating that under this situation, BEDROC can no longer address the "early recognition" problem. While the value of α was increasing, the linear correlation of BEDROC and AUC-ROC became less obvious (**Figure 7, B, C & H, I**).

As BEDROC focuses on the earlier part of the ranking, the Pearson coefficients of BEDROC and AUC-ROC calculated from Vina's result dropped in a linear-like trend (**Figure 7, J**), whereas, in LIDAEUS's result, the coefficients dropped drastically when α was increased from 20 to 160.9 (**Figure 7, D**), which might be due to LIDAEUS' incapability of addressing the "early recognition" problem and the approach used in generating the ranking lists from the results of LIDAEUS. Given the occurrence of ligand duplicates in the outputs of LIDAEUS, compounds that were not shown in results, were manually added in the end of final ranking list with the artificial energy scores. Since BEDROC is still dependent on the ranking, such artificial error should

affect the integrity of the data, however, AUC-ROC can possible be less influenced because ranking of the same score was treated as random distribution.

Due to EF only focusing on the “early recognition” problem, the high Pearson coefficients derived from EF and BEDEOC (**Figure 7, E, F & K, L**) indicates that BEDROC can indeed measure VS tools’ capabilities of addressing the “early recognition” problem. Nevertheless, owing to the absolute error in LIDAEUS’s ranking list, BEDROC is unsuitable to evaluate LIDAEUS’s performance in this project. The aggregation of a large number of points of 0 in **Figure 7 (E & F)** suggests that LIDAEUS has a dissatisfactory performance on ranking actives on the of early part of the output, where Vina owns a clear ascendancy (**Figure 7, K & L**). Thereby, AUC-ROC and EF were employed in the subsequent evaluation of the performance of both LIDAEUS and Vina.

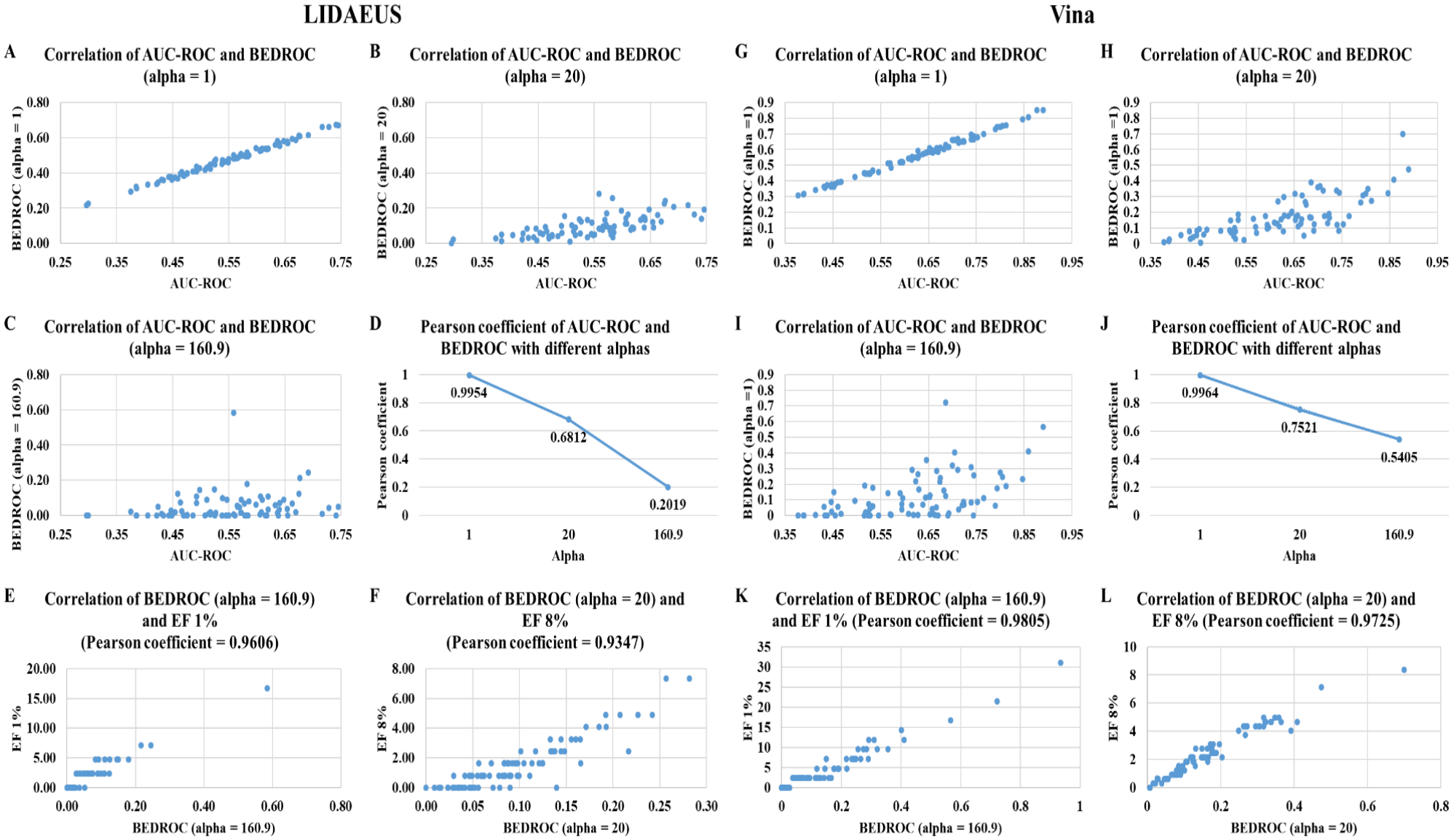


Figure 7. Correlation of different metrics from the result of LIDAEUS and Vina. AUC-ROC: the probability of an active with higher score than a decoy in random selection. **BEDROC**: when more actives are shown in the range of interest and ranked before decoys, a higher value is obtained. **EF**: when more actives are shown in the range of interest, a higher value is obtained. **A & G**: AUC-ROC vs. BEDROC (α = 1; top 80% of the ranking contributes to 80% of the total BEOROC score). **B & H**: AUC-ROC vs. BEDROC (α = 20; top 8% of the ranking contributes to 80% of the total BEOROC score). **C & I**: AUC-ROC vs. BEDROC (α = 160.9; top 1% of the ranking contributes to 80% of the total BEOROC score). **D & J**: Pearson coefficients calculated from AUC-ROC and BEDROC with different values of α. **E, F & K, L**: Correlation of EF and BEDROC measuring the same ranges, the top 1% and the top 8% of the ranking respectively.

Comparison between LIDAEUS and Vina

Overall performance evaluation

Given the relatively small scale of benchmarking ligand data compared to the amount of compounds in real-world screening but enormous amount of tagets for mannual analysis, attention was paid to the overall performance comparison rather than the “early recognition” problem and the performance of VS programs was grouped into four grades depending on the AUC-ROC value obtained from the screening results against targets. Therefore, AUC-ROC was used as the major metirc and EF was employed to compare specific examples. (Detailed ROC graph and AUC-ROC of each target in **Appendix 5 & 7**). **Figure 8** shows that Vina outperformed LIDAEUS regardless of the input ligand data used by LIDAEUS, however, using ligand multiple conformers can indeed improve the performance of LIDAEUS, which increased the number of targets that LIDAEUS can achieve “**distinction**” performace from 4 to 8 (**Figure 8, B & C**).

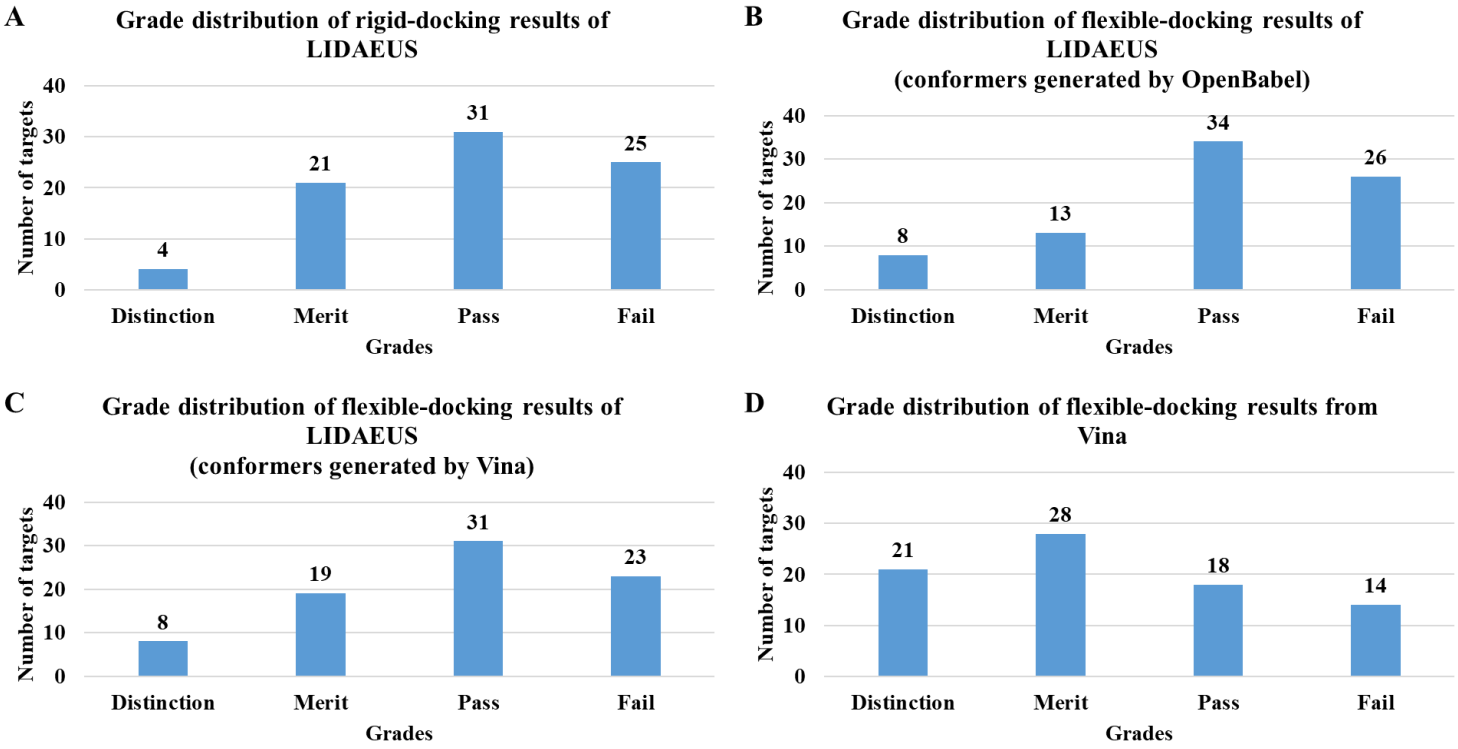


Figure 8. Grade distribution according to the value of AUC-ROC. Distinction: AUC-ROC greater or equal to 0.70; **Merit**: AUC-ROC less than 0.70 and greater or equal to 0.60; **Pass**: AUCOC less than 0.60 and greater or equal to 0.50; **Fail**: AUC-ROC less than 0.50. **A**: LIDAEUS used ligand single-conformers provided by DEKOIS 2.0 set as input ligand data. **B & C**: LIDAEUS performed better when using ligand multiple conformers. Notably, conformers generated by Vina played a more successful role in this improvement than the ones generated by OpenBabel did, which not only increased the number of “distinction” but also decreased the number of “fail”. **D**: although Vina obtain “distinction” in 21 targets, yet, it still performed worse than random selection in 14 targets, owning the “**fail**” performance.

Predominant examples comparison

Figure 9 shows the best examples of LIDAEUS using different ligand input data and Vina conducting flexible docking. When LIDAEUS conducted rigid docking, Vina outperformed LIDAEUS in any of its predominant examples except ER-beta, where LIDAEUS generated results with EF 1% of 16.69 out of 31.00 but the result of Vina failed to produce this EF 1% value. Notably, regardless of the input data, LIDAEUS obatain such EF 1% value consistently when screening against ER-beta. But, still, the ER-beta result of Vina obtained AUC-ROC value of 0.63 indicating a better overall performance. In this comparison, Vina enjoyed a unparalleled position in addressing the "early recognition" problem. In the screening result of Cyclooxygenase 2 (**COX-2**) generated by Vina (**G in Figure 9, α, β & γ**), compounds rank before the top 1% were all active ligands, that is, 12 active molecules, while the best performance of LIDAEUS docking against such target was placing 1 active compound in the top 1% conducting flexible docking with conformers generated by OpenBabel (**H in Figure 9, β**).

In the screening against ERBB2, although the results of LIDAEUS and Vina had similar AUC-ROC values (**A & B in Figure 9, α**), LIDAEUS failed to place any active ligand in the top 1% of the total compounds in its ranking list, whereas, there were 4 actives ranking before the top 1% in Vina's result where the EF 1% was 11.69 out of 31.00. Nevertheless, the AUC-ROC values of LDAEUS's best results rarly exceeded 0.80 while the best examples of Vina not only obtained AUC-ROC values that never appeared in LIDAEUS's best results, 0.89 and 0.88, but also produced high EF 1% and EF 3%.

Overall, LIDAEUS was still able to produce results with fairly satifactory AUC-ROC values, leading to the "distiction" performance grade (**A in Figure 9, α, β & γ**), and in some cases (**A & B in Figure 9, β**), LIDAEUS outperformed Vina regardless of the metrics. However, for addressing "early recoginition" problem, all screenings carried out with LIDAEUS were failures expect the result of ER-beta.

Note: the maximum EF 1% and EF 3% for each target both are 31.00. The ratio of actives and the total compounds in the range of interest is the ratio of obtained EF and the maximum EF.

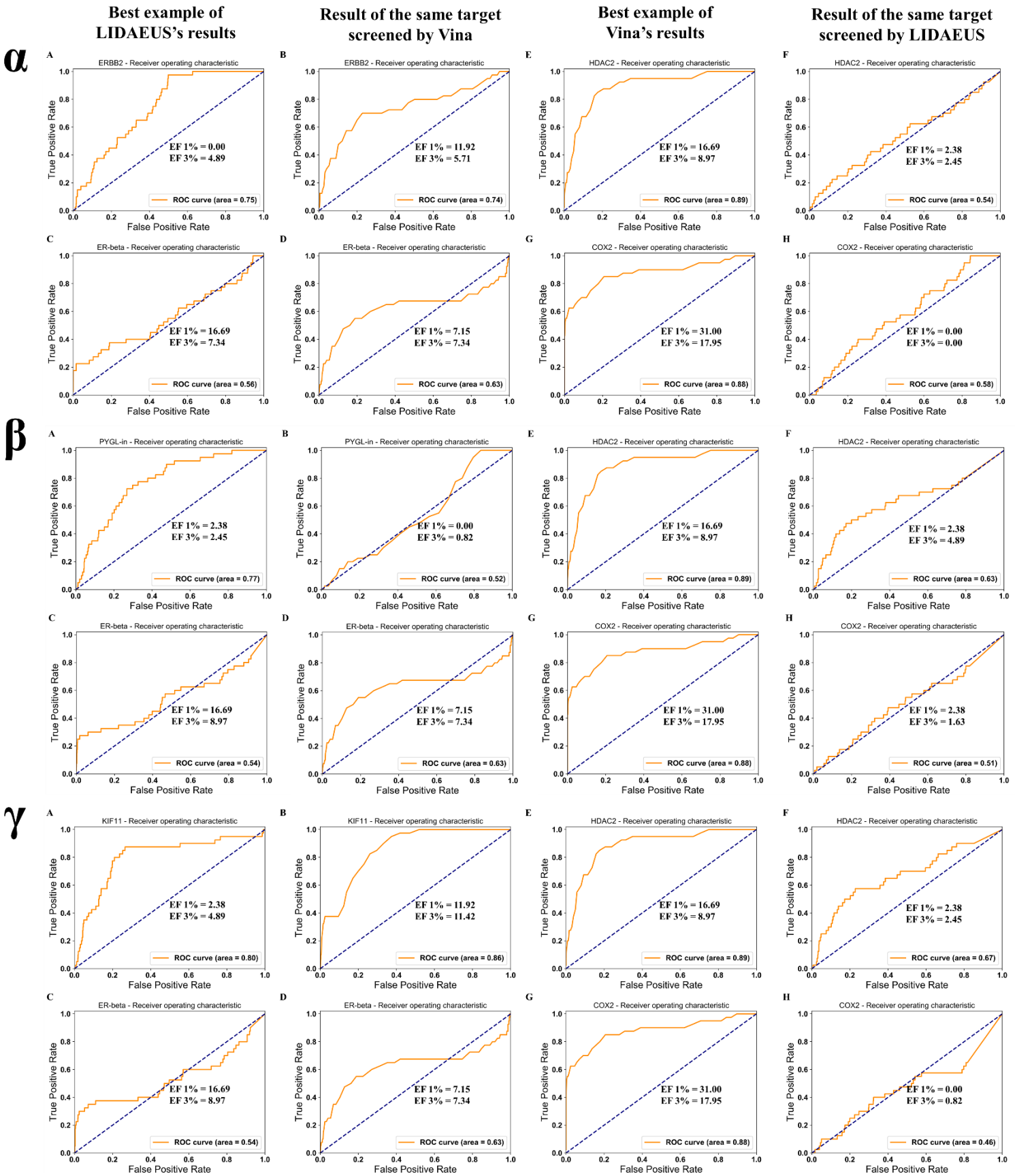


Figure 9. (Excerpted from Appendix 6)

α: Example comparison of LIDAEUS's rigid-docking performance and Vina's flexible-docking performance. A & **E**: the best examples in LIDAEUS and Vina respectively according to the AUC-ROC values. **C** & **G**: the best examples in LIDAEUS and Vina respectively according to the EF 1% and EF 3%.

β: **Example comparison of LIDAEUS's flexible-docking performance (conformers generated by OpenBabel) and Vina's flexible-docking performance. A** & **E**: the best examples in LIDAEUS and Vina respectively according to the AUC-ROC values. **C** & **G**: the best examples in LIDAEUS and Vina respectively according to the EF 1% and EF 3%.

γ: **Example comparison of LIDAEUS's flexible-docking performance (conformers generated by Vina) and Vina's flexible-docking performance. A** & **E**: the best examples in LIDAEUS and Vina respectively according to the AUC-ROC values. **C** & **G**: the best examples in LIDAEUS and Vina respectively according to the EF 1% and EF 3%.

Attempts to improve LIDAEUS's performance

Based on above performance comparison, possible improvements made to LIDAEUS were focusing on increasing the AUC-ROC values derived from LIDAEUS's results.

Result-lead optimization experiment

By changing the parameters of sitepoint generation, the performance of LIDAEUS were successfully improved in 24 out of 25 targets where the AUC-ROC of LIDAEUS's results fell into the "**fail**" area (**Figure 12**).

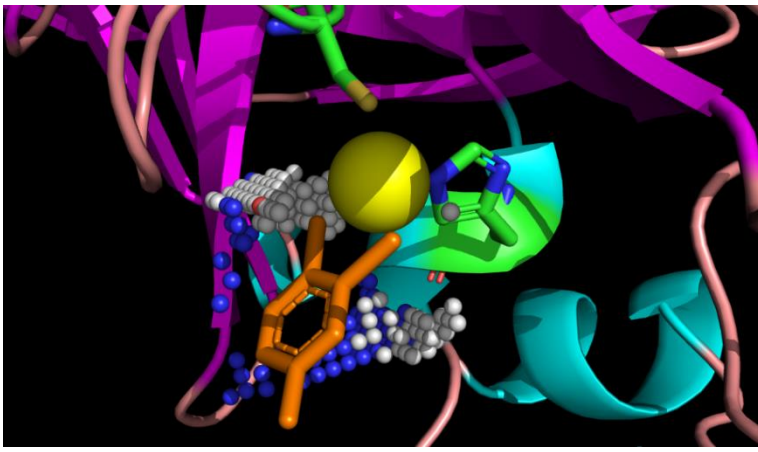


Figure 10. Snapshot of the active site of SARS-HCoV in complex with 4-methylbenzene-1,2-dithiol (TDT) and sitepoints generated by LIDAEUS. TDT is colored in orange. The zinc atom is colored in yellow and shown as a sphere. Key amino residues interacting with the zinc atom was in bold and green. Each sitepoint is colored depending on built-in ligand-protein interaction. (Hydrophobic: white; HBD: blue; HBA: red)

The reason why improvements failed to be made to LIDAEUS when docking against SARS-HCoV is that the zinc atom in this active site (**Figure 10**) plays a pivotal role in the interaction with the co-crystallized ligands (TDT), however, it was discarded in the processing of target structure as such interaction was not recorded in LIDAEUS and the process of either fitting ligands onto the sitepoints or calculating energy in LIDAEUS was only based on interaction of van der Waals and hydrogen bonding. Notably, although LIDAEUS's performance were improved from AUC-ROC of 0.30 to AUC-ROC of 0.57 (**Figure 12**), the optimized parameters had significant difference from parameters used in other targets' sitepoint generation, where the hydrogen bonding energy cut-off was extremely low (-1.5 for HBD and HBA energy). Such setting resulted in the generation of sitepoints (Optimized sitepoints) consist of 49.65% hydrogen bonding donor atoms (**Figure 11, B; Table 5**), whereas, the sitepoints (Default sitepoints) generated using built-in parameters were more hydrophobic, where the hydrophobic atoms accounts for 89.21% of total atoms (**Figure 11, A; Table 5**). Sitepoints (**Figure 11, C**),

inheriting the atom type from the Default sitepoints and the atom distribution from the Optimized sitepoints, were also used in screening against RXR, the result of which produced an AUC-ROC of 0.57, the same as the one from the optimized result (**Figure 12**). Such outcome indicates that the atom distribution or the shape of the sitepoints may play a significant role in the docking process of LIDAEUS.

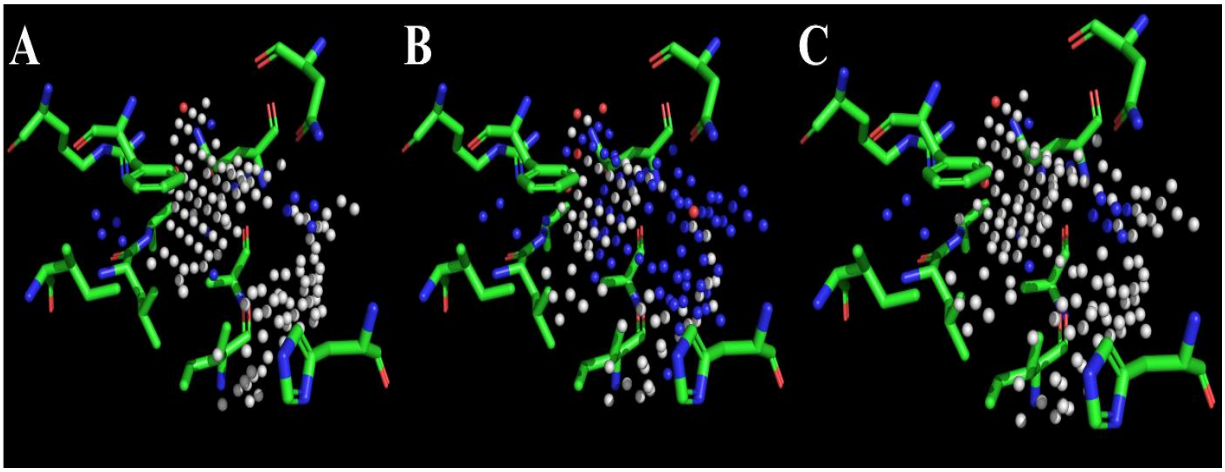


Figure 11. Sitepoints generated under default parameters (**A)**, optimized parameters (**B**) and sitepoints (**C**) with **A-like** atom components but in distribution like **B**. Key amino residues interacting with co-crystal ligand (not shown) is extracted and shown in bold. Each sitepoint is coloured depending on built-in ligand-protein interaction. (**Hydrophobic**: white; **HBD**: blue; **HBA**: red)

Setting of Sitepoint generation	Hydrophobic (%)	HBD (%)	HBA (%)
Default parameters	89.21	9.35	1.44
Result-lead Optimized parameters	46.85	49.65	3.50
Atom-type-altered	88.81	9.79	1.40

Table 5. Component analysis of sitepoints. Each atom definition is according to the built-in setting of LIDAEUS. **Hydrophobic**: hydrophobic atoms including sp^1, sp^2, sp^3 and aromatic carbon. **HBD**: hydrogen bond donor atoms including sp^3 oxygen, sp^3, sp^4 and ammoniacal nitrogen. **HBA**: hydrogen bond acceptor atoms: sp^2, sp^3 and carbonyl oxygen.

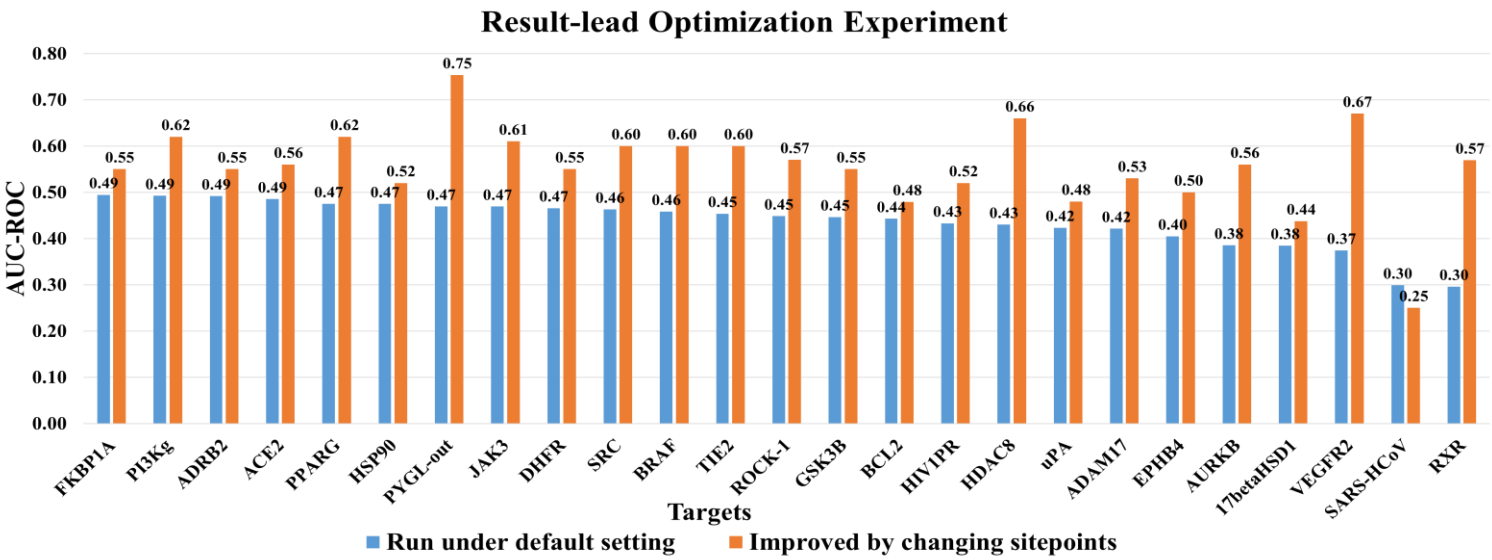


Figure 12. AUC-ROC comparison between results from running LIDAEUS under default setting and results from optimization experiments. Result-lead Optimization Experiment: the changes made to the parameters of LIDAEUS' sitepoint generation system was dependent on whether the AUC-ROC can be increased by the previous changes. **PYGL-out**: an allosteric site of Glycogen Phosphorylase L. **VEGFR2:** vascular endothelial growth factor. The improvement made in the screening against **VEGFR2** is impressive, which increased the AUC-ROC from 0.37 to 0.67, whereas, changing the parameters of sitepoint generation failed to optimize the screening results of SARS-HCoV. Among 25 targets, LIDAEUS only succeeded in obtaining "distinction" performance in docking ligands against **PYGL-out**.

Modification on docking fitness tolerance & number of sitepoints

Investigation into the impact of docking fitness tolerance (tol) and the number of sitepoints (spopt) on the VS performance of LIDAEUS was carried out by respectively varying these two parameters throughout multiple dockings against RXR. Such experiments reveal that both spopt and tol variance can indeed ameliorate the VS performance of LIDAEUS comparing to using the default values, but exponential incensement in computational time also occurred in these experiments (**Figure 13, A & B**). In the tol variance experiment, LIDAEUS reach the performance plateau in an acceptable scale of computational time, that is about 5000s (**Figure 13, B**), whereas, it took approximate 10000s for LIDAEUS to obtain its best performance in experimenting spopt variance. Nevertheless, after the performance peak, LIDAEUS started to perform worse and worse (**Figure 13, A**).

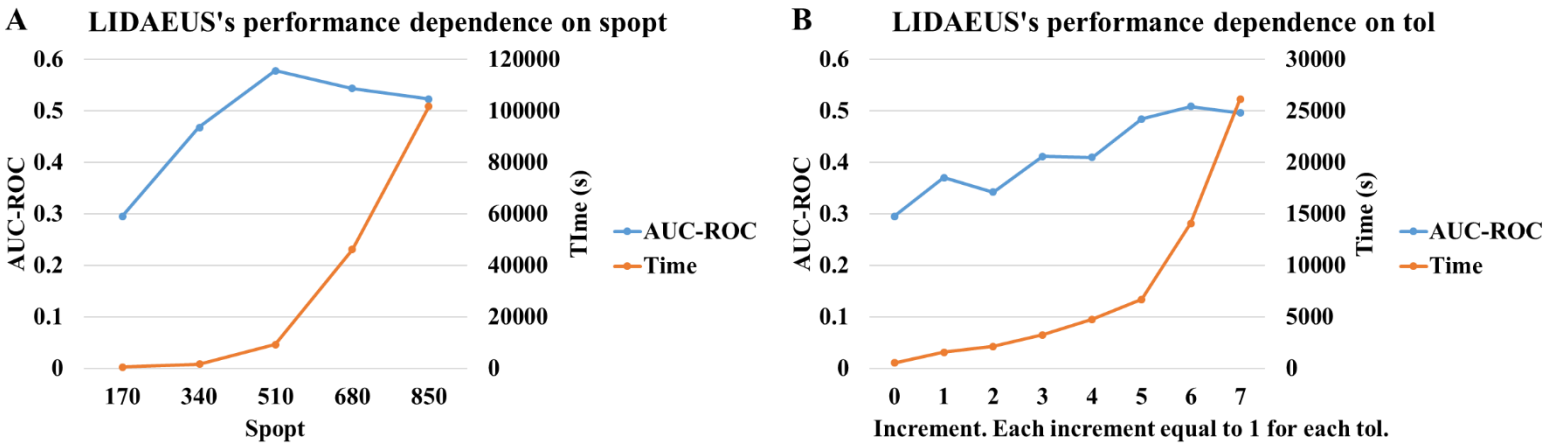


Figure 13. Impact of docking fitness tolerance (tol) and the number of sitepoints (spopt) on the VS performance of LIDAEUS. A: each increment of spopt was set to 170. LIDAEUS obtain its best performance when spopt reach 510 and only after that, LIDAEUS started to perform better than random selection. However, LIDAEUS's performance also declined when spopt exceeded 510. **B**: tol consists of three parts. Increment was made to each part of tol. Only when 6 increments were made to tol, did LIDAEUS reach its best performance.

Random re-shaping sitepoints

Given the limited computing resource, re-shaping sitepoints was only carried out on the docking experiments of RXR and ERBB2. The XYZ coordinates of a sitepoint generated under default setting were respectively added or deducted a random number less than 1. **Figure 14 (A)** shows that such method succeeded in making improvement to the performance of LIDAEUS docking against RXR. Even though some improvements were huge, increasing the AUC-ROC up to 0.58, nonetheless, 86% of the sitepoints resulted in "**fail**" performance of LIDAEUS. Still, this random re-shaping approach is able to make LIDAEUS produce better screening results even in ERBB2, the target that LIDAEUS performed the best when conducting default rigid docking (**Figure 14, C**). However, such method is unsuitable for being employing in blind VS because of its huge uncertainty. In the experiment of re-shaping ERBB2's sitepoints, a significant portion of results fell into ranges where AUC-ROC varied from 0.51 to 0.73 (**Figure 14, C**), essentially lowering the docking performance of LIDAEUS. Despite great improvements appearing in RXR experiment, the probability of LIDAEUS performing better than random selection is only 14%.

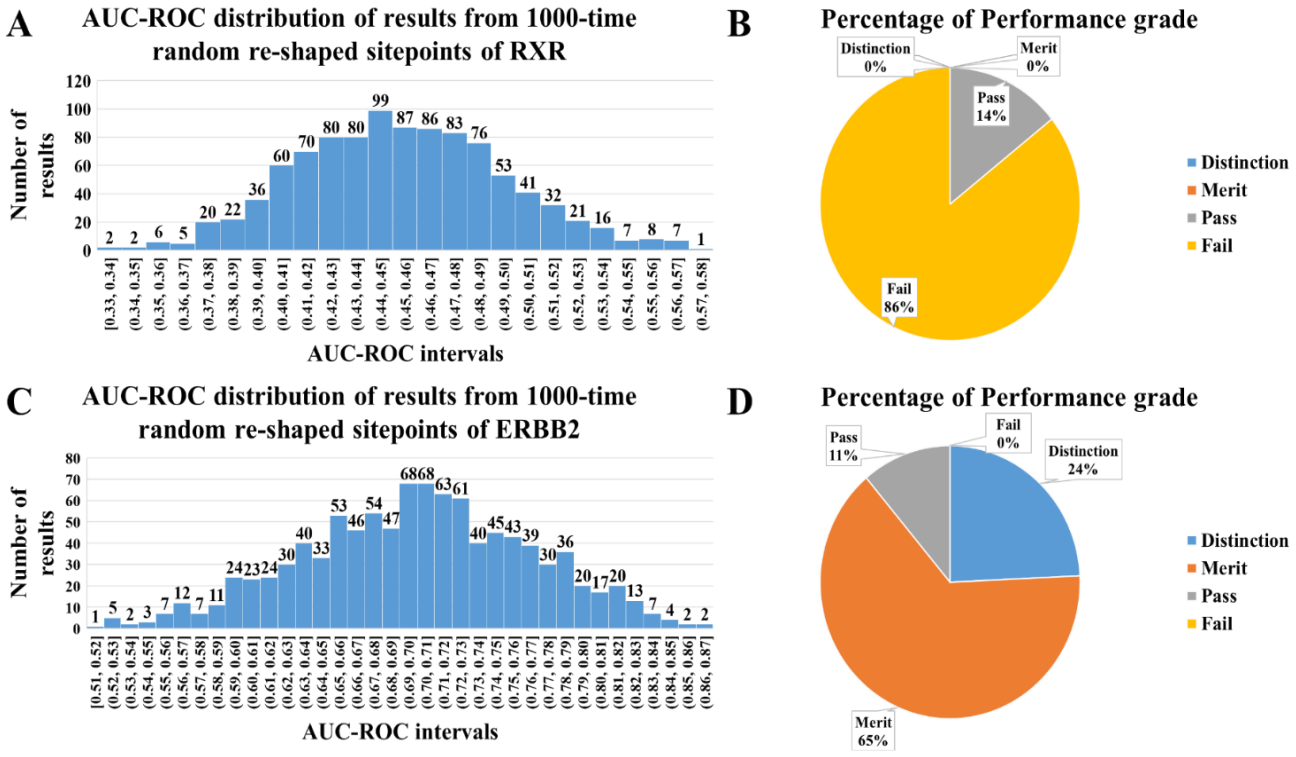


Figure 14. Histogram of AUC-ROC distribution derived from 1000-time random re-shaping sitepoints (**A** & **C) and associated performance grade analysis (B** & **D)**. In **RXR**, overall, every re-shaped sitepoints generated results with higher AUC-ROC than the Default sitepoints did, whereas, in **ERBB2**, only partial results have higher or equivalent AUC-ROC compared to the AUC-ROC (0.74) derived by using sitepoints generated under default setting. The AUC-ROC distribution in the example of RXR behaves more like a normal distribution, showing more steady improvements. The variance in the ERBB2 experiment indicates that this approach performs more unstably in such target.

Fused sitepoints

Multiple dockings against RXR and ERBB2 were performed with LIDAEUS by using fused sitepoints derived by combining 2 out of 1,000 sets of random re-shaped sitepoints. **Figure 15** shows that such approach succeeded in improving LIDAEUS's VS performance. Although the probability of obtaining the best performance is small, regardless of target, improvement can be easily obtained with relatively cheap computational expense compared to randomly re-shaping sitepoints.

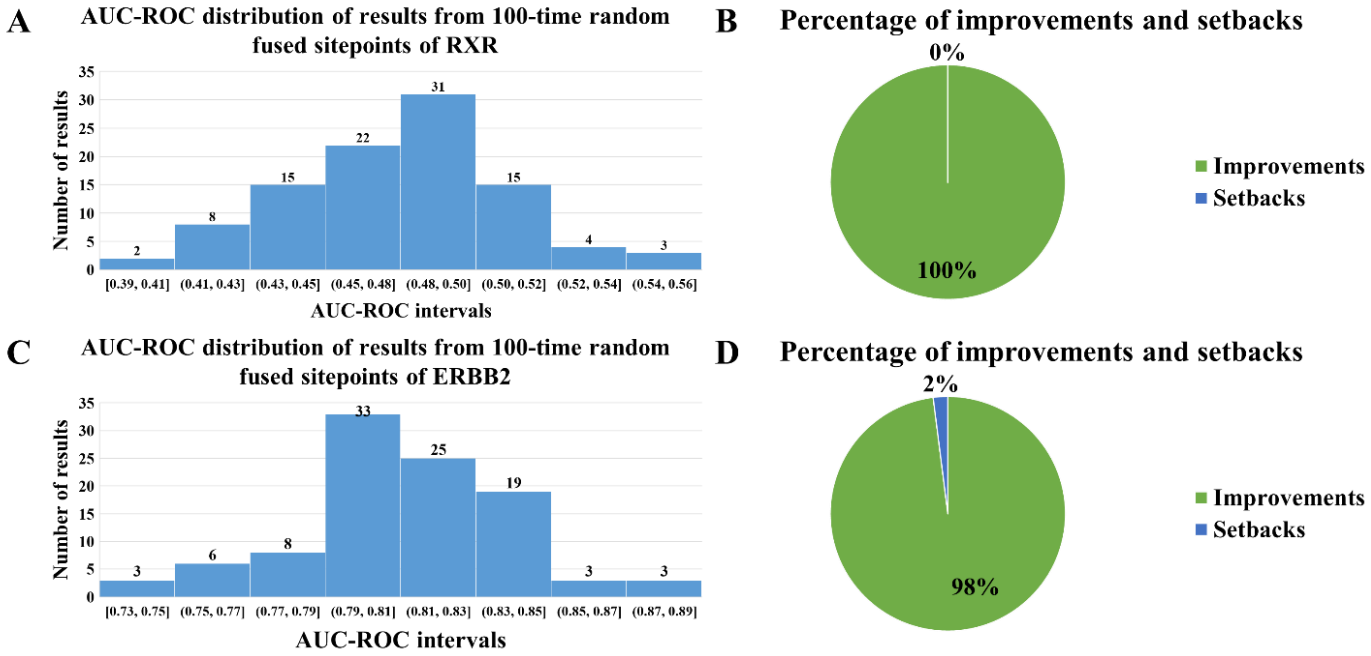


Figure 15. Histogram of AUC-ROC distribution derived from 100-time random fused sitepoints (A & C) and associated improvements and setbacks analysis (B & D). In **RXR**, every set of fused sitepoints resulted in improvements and approximate 22% of the results produced AUC-ROC over 0.50. The average time of docking for each set of sitepoints is 850 s. In **ERBB2**, 98% of the results made improvements to LIDAEUS's VS performance, among which, the number of sitepoint sets leading to over 0.80 AUC-ROC accounted for over 33%. The average time of docking for each set of sitepoints is 1659 s.

Re-scoring

Ligand poses in the results of LIDAEUS's rigid-docking were re-scored by using four different PIP score calculations. Each target's result was evaluated by using AU-ROC. Still, previous performance grades were employed to evaluate LIDAEUS's VS performance over 81 targets. Overall, using PIP scores as the reference to rank ligands can improve LIDAEUS's performance with increasing the number of "**distinction**" performance (**Figure 16**). Ranking by **PIP_4** score successfully made LIDAEUS obtain "**distinction**" in up to 8 targets while decreasing the number of "**fail**" performance. Although LIDAEUS achieved the least "**merit**" using **PIP_3** score for ranking, the "**fail**" performance appear the least frequently with such score.

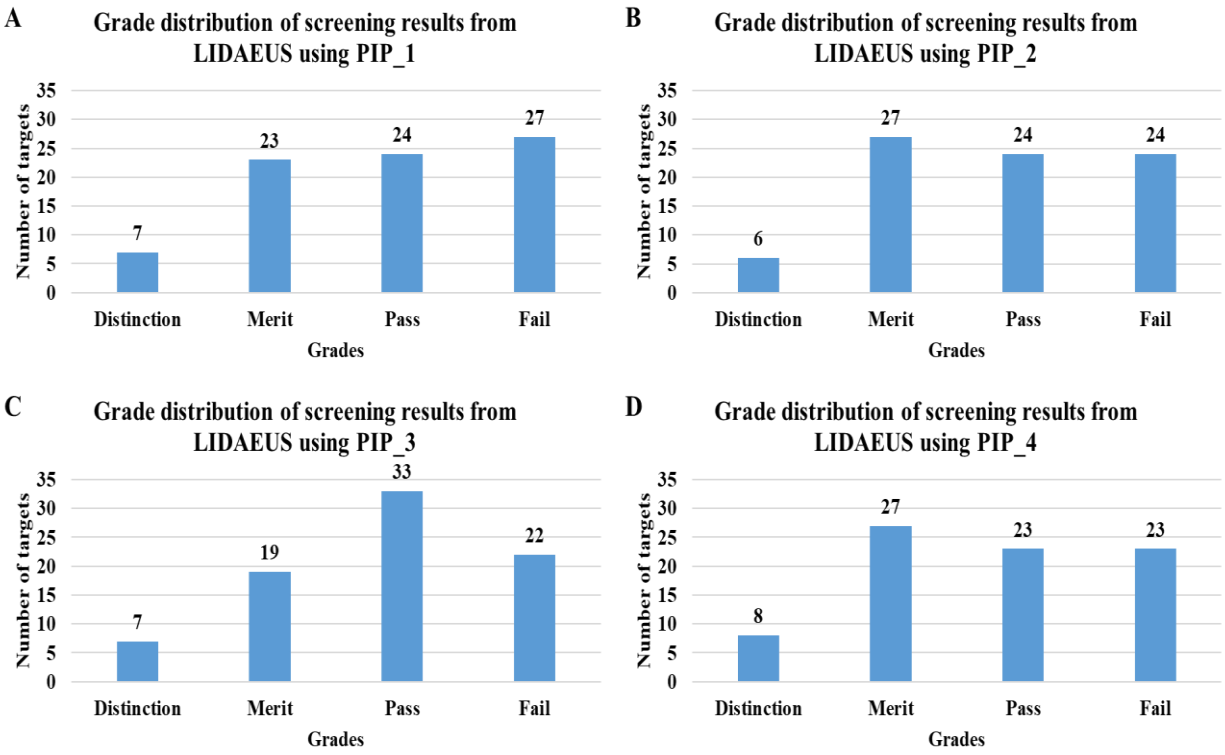


Figure 16. Performance grade distribution of the results derived by using different PIP scores. PIP score: Tanimoto coefficients calculated depending on whether docked ligand and the target have the same interaction as reference ligand and the target have. **PIP_1** & **PIP_2**: atom level comparison. **PIP_1** allows the occurrence of atoms that do not appear in the reference interaction while **PIP_2** only allows the occurrence of the key atoms. **PIP_3** & **PIP_4**: residue level comparison. In **PIP_3**, docked ligand just need contacts with key residues while **PIP_4** requires docked ligand interact with the key atoms of the key residues.

PIP scores were employed to increase the energy scores of ligand poses matching the criteria, however, no apparent improvement was revealed in **Figure 17**. Such re-calculation made no difference from the original energy calculations. Different weights were then assigned to three enthalpy energetic terms, namely, **vDW**, **HBD** and **HBA**, in order to investigate suitable energy calculation for each target. Same method was also used in exploration of suitable weights of enthalpy and entropic terms (**Figure 18**).

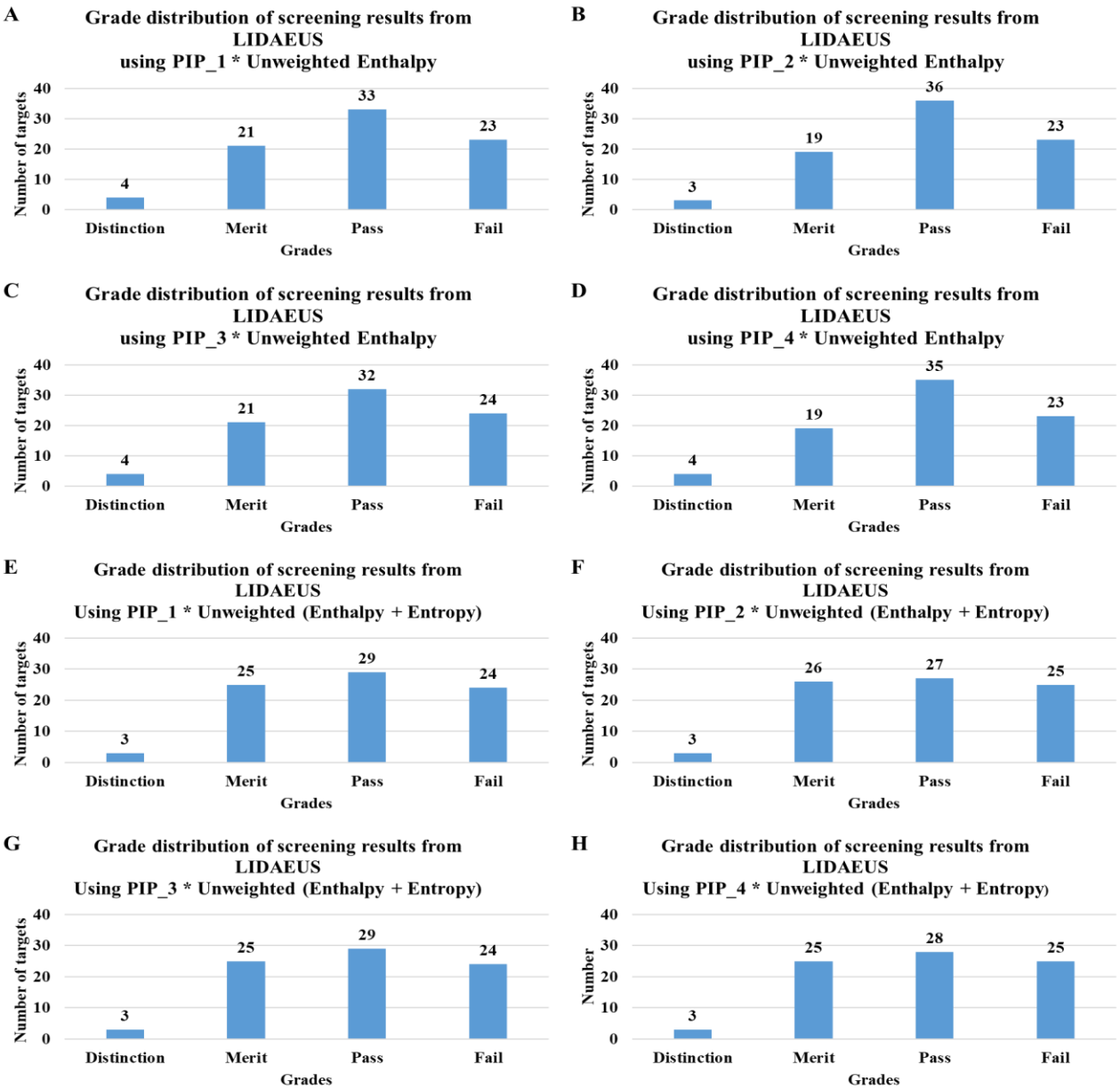


Figure 17. Different unweighted energy scores combined with four PIP scores respectively. A, **B**, **C** and **D**: performance grade distribution of experiments using different PIP scores multiply unweighted enthalpy. Essentially, no improvement was made by using such energy re-calculation approaches although the number of "fail" performance dropped one or two among the 4 calculations. **E**, **F**, **G** and **H**: performance grade distribution of experiments using different PIP scores multiply unweighted enthalpy and unweighted entropy. Here, the entropic term calculation was derived from a simple water loss counting and empirical side chain conformational entropy loss calculation (Code outlined in **Appendix 8**). Methods used in **E**, **F**, **G** and **H** worsened the VS performance of LIDAEUS by decreasing the number of "**distinction**" performance.

Figure 18 (B) reveals the introduction of unweighted simple entropic terms worsened LIDAEUS's performance. However, when optimized weights and suitable PIP scores were employed in the screening against each target, the performance of LIDAE US was hugely improved (**Figure 18, C, D, E & F**).

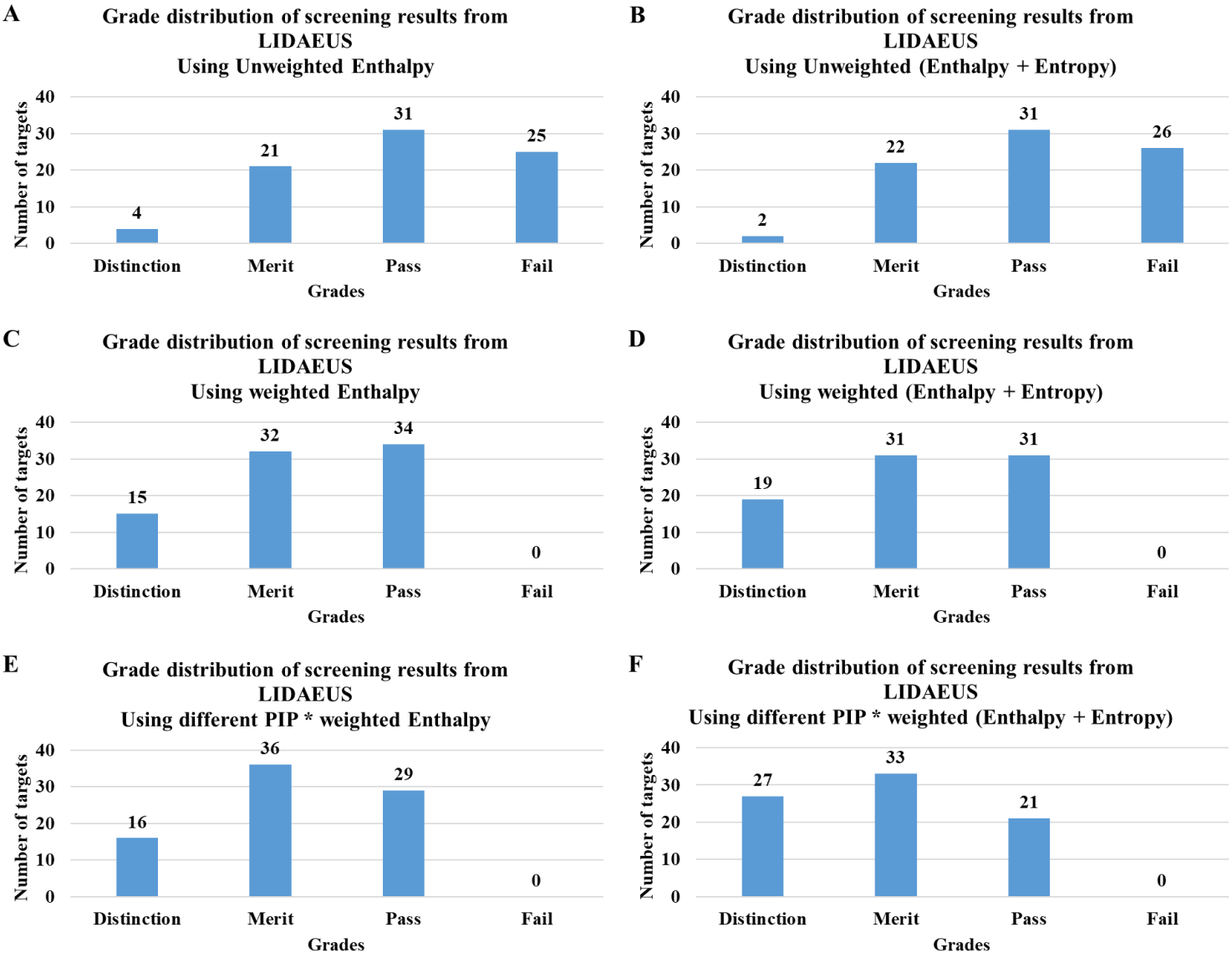
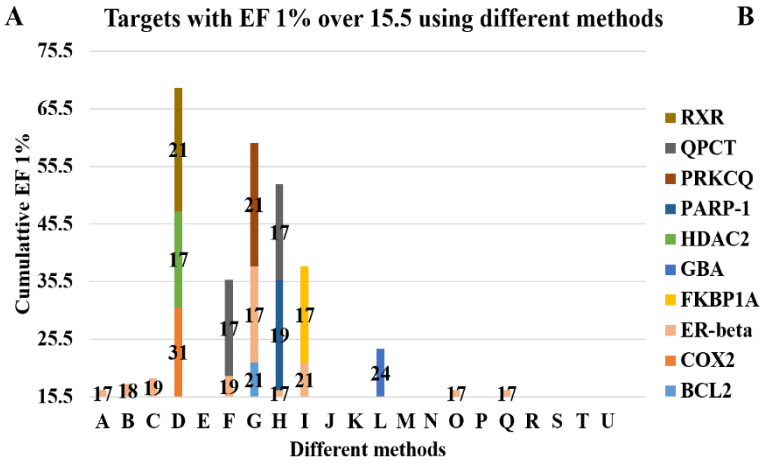


Figure 18. Weighted energy calculation. A: performance grade distribution of experiments using the original enthalpy calculation method. **B**: performance grade distribution of experiments using the original enthalpy in addition with simple entropic terms. **C**, **D**, **E** and **F**: performance grade distribution of results where suitable energy weights and PIP scores were employed in associated methods for each target. After such target-lead optimization, the performance of LIDAEUS was remarkably ameliorated. Notably, **F** shows a better LIDAEUS overall performance than Vina' one in **Figure 8 (D)**. **Note**: the absence of "fail" performance in **C**, **D**, **E** and **F** was due to weights or PIP scores equal to 0, leading to all compounds sharing the same scores, which was deemed as random selection by the AUC-ROC calculation. In other words, LIDAEUS cannot perform better than a random selection among those targets. Suitable weights were selected depending on whether resulting outcome producing the best AUC-ROC values.

EF 1% were calculated from results generated by multiple methods including using different ligand input data or re-scoring methods in LIDAEUS and default flexible docking method in VINA. **Figure 19 (A)** shows that LIDAEUS possesses a capability of enrich actives when screening against ER-beta, regardless of various methods employed. Undoubtedly, Vina still outperformed LIDAEUS in enriching actives, however, compared to the results derived from LIDAEUS's default method, obvious progresses can be observed when re-scoring methods were employed (**Figure 19, A**), where LIDAEUS not only succeeded in addressing active enrichment problem in more targets but also enriched more actives in the top 1% of the total compounds for each targets.



B

Label	Method	Label	Method	Label	Method
A	Rigid docking	H	PIP * weighted Enthalpy	O	PIP_2 * unweighted Enthalpy
B	Flexible docking (conformers generated by OpenBabel)	I	PIP * weighted (Enthalpy + Entropy)	P	PIP_3 * unweighted Enthalpy
C	Flexible docking (conformers generated by Vina)	J	PIP_1	Q	PIP_4 * unweighted Enthalpy
D	Vina's flexible docking	K	PIP_2	R	PIP_1 * unweighted (Enthalpy + Entropy)
E	Unweighted (Enthalpy + Entropy)	L	PIP_3	S	PIP_2 * unweighted (Enthalpy + Entropy)
F	Weighted Enthalpy	M	PIP_4	T	PIP_3 * unweighted (Enthalpy + Entropy)
G	Weighted (Enthalpy + Entropy)	N	PIP_1 * unweighted Enthalpy	U	PIP_4 * unweighted (Enthalpy + Entropy)

Figure 19. EF 1% calculation across different methods. EF 1% over **15.5** means there were more than 6 active molecules in the top 1% of the total compounds. Methods that generated result with EF 1% over 15.5 in more targets obtained higher cumulative EF 1%. **Note**: methods not denoted Vina were employed in LIDAEUS docking or re-scoring. The maximum EF 1% is 31 for each target. The ratio of actives and the total compounds in the range of interest is the ratio of obtained EF and the maximum EF. In methods **E**, **F**, **G**, **H** and **I**, weights were selected depending on whether resulting outcome producing the highest EF 1%.

Large-scale screening

A large-scale screening was performed with LIDAEUS against ER-beta using a decoy data set of 54,028 compounds, where 2 molecules were eliminated because of similar physicochemical features to the one of active ligands. The first 1,000 compounds were subsequently screened by Vina in order to simulate the workflow of employing LIDAEUS and Vina in real-world project. Based on this, the number of active compounds in the range of interest became the major metric in this experiment. **Table 6** reveals that LIDAEUS indeed possesses the ability to address the “early recognition” problem in the screening against ER-beta, placing 9 out of 40 active compounds in the top 1,000 compounds and 3 out of 40 actives in the top 40 compounds. Subsequent screening experiment conducted with Vina shows great enrichment of actives. 6 out of 9 active molecules entered the top 10% of the total molecules, which is the first 100 compounds. Such active enrichment is quite a success in a real-world project.

VS program	AUC-ROC	Number of actives					Number of total actives	Total compounds
		Top 0.074%	Top 1%	Top 1.8%	Top 3%	Top 10%		
LIDAEUS	0.59	3	7	9	9	11	40	54,066
Vina	0.91	0	2	3	3	6	9	1,000

Table 6. Actives enrichment in real-world VS screening simulation. Top 1.8% of LIDAEUS’s screening result represent the top 1,000 compounds.

Binding site clustering

In our project, a binding site was defined by the amino acid residues components around both the original co-crystalized ligand and crystal-structure-available ligands from DEKOIS 2.0 set within 3.5 Å for a target. The binding sites of targets were successfully extracted except the site of target SARS-HCoV where no amino acid residues appeared in 3.5 Å.

The analysis of the components of binding sites was successively carried out by calculating the percentage of amino acids with hydrophobic residues, the percentage of amino acids only with hydrophobic atoms in

residues, the cumulative hydrophobicity of each binding site (Related definition in **Appendix 2**) and the atom-type-based geometry descriptors. Hierarchical clustering using average distance based on above 4 features were carried out.

Figure 20 (A, B, C & D) reveals that no consistent LIDAEUS VS performance were rarely observed from sub-clusters. In **Figure 20 (A)**, although targets with high AUC-ROC results, ERBB2 and tissue plasminogen activator (**TPA**), were grouped together, in such group, LIDAEUS had only "**pass**" or "**fail**" performance when screening against the other members. Similar situation happened to other targets of high AUC-ROC values when using the percentage of hydrophobic amino acids as clustering reference. While clustering by using the percentage of amino acids with hydrophobic atoms in residues (**Figure 20, B**), in the cluster of proto-oncogene tyrosine-protein kinase Src (**SRC**), glucocorticoid receptor (**GR**), Thrombin, purine nucleoside phosphorylase (**PNP**) and histone deacetylase 8 (**HDAC8**), LIDAEUS obtain "**merit**" performance in 3 out of 5 group members, however, still, "fail" performance occurred in the screening against the 2 left targets. A successful clustering was performed with cumulative hydrophobicity (**Figure 20, C**). Inhibin alpha (**INHA**), acetylcholinesterase (**ACHE**) and phosphodiesterase 4B (**PDE4B**) were the exclusive 3 members of a group, against which LIDAEUS all obtained "**merit**" performance.

By contrast, atom-type-based geometric clustering succeeded in grouping three targets of low AUC-ROC values, which are SRC, rho-associated protein kinase 1 (**ROCK-1**) and phosphoinositide 3-kinase gamma (**PI3kg**) (**Figure 20, D**). Surprisingly, this method also managed to generate a cluster of interleukin-2-inducible T-cell kinase (**ITK**), cyclooxygenase-1 (**COX-1**), serine/threonine-protein kinase B-Raf (**BRAF**), PNP and serine/threonine-protein kinase Pim-2 (**PIM-2**), where LIDAEUS performed "**merit**" in 4 out of 5 targets although the "fail" performance still occurred in screening against BRAF.

Overall, multiple clustering shows that the instability of LIDAEUS's VS capability renders performance consistency difficult to obtain in the screening against any families of targets.

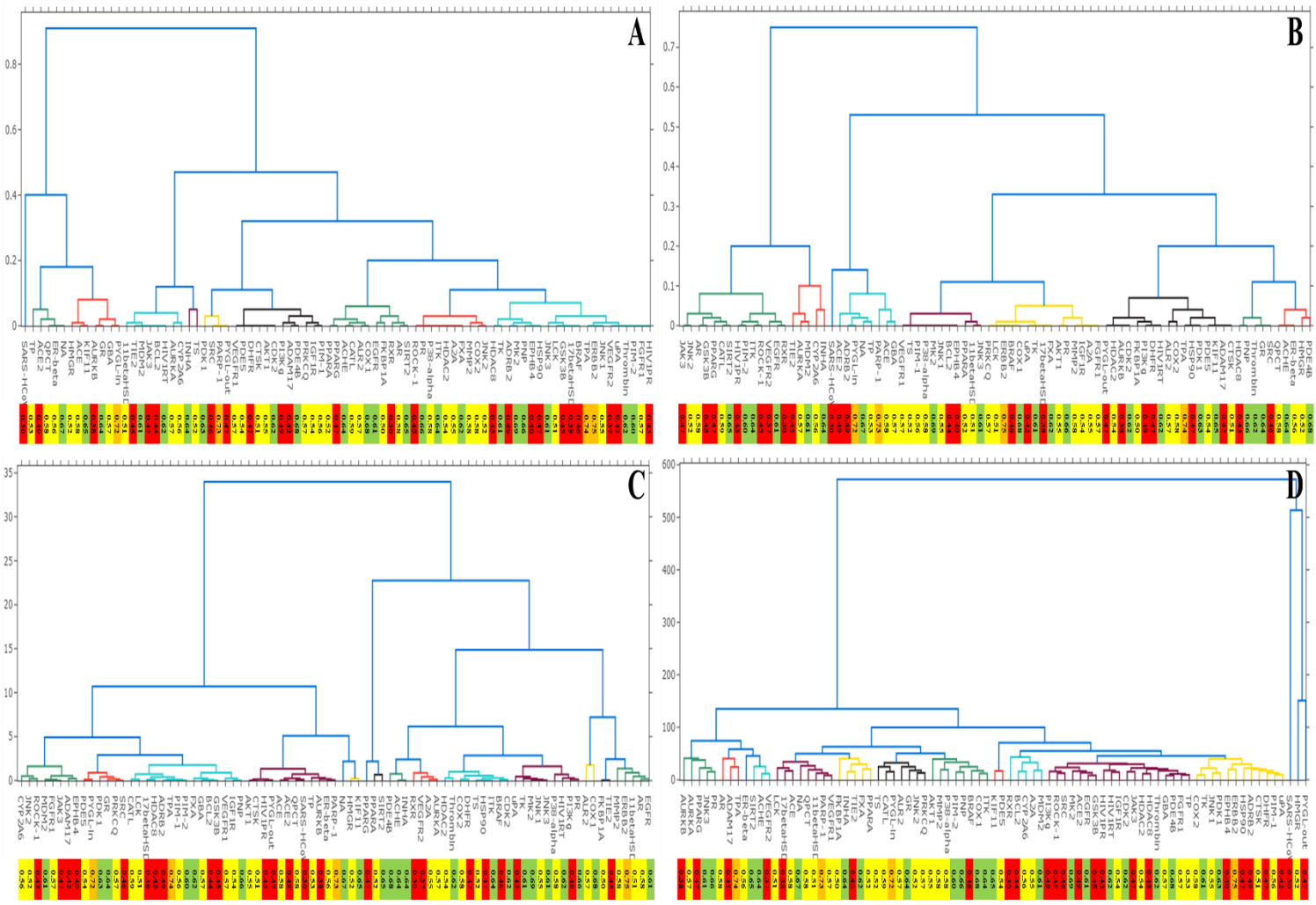


Figure 20. Hierarchical clustering using average distance based on different descriptors. A: cluster binding site according to the percentage of amino acid with hydrophobic residues of their components. **B**: cluster binding site according to the percentage of amino acid with hydrophobic atoms in residues. **C**: cluster binding site according to the cumulative hydrophobicity of each binding site. **D**: cluster binding site according to the atom-type-based geometric descriptors. Due to only zinc atom in the binding site, SARS-HCoV was isolated under any clustering methods except using cumulative hydrophobicity. **Note**: the number inside the color bar represents the ROC-AUC value obtained from the screening against corresponding target (extracted from **Appendix 5**). Different color scale indicates different range of AUC-ROC value. **Orange**: AUC-ROC value over or equal to 0.70. **Green**: AUC-ROC value less than 0.70 but over or equal to 0.60. **Yellow**: AUC-ROC value less than 0.60 and greater or equal to 0.50. **Red**: AUC-ROC value less than 0.50.

Discussion

As was stated before, employing structure-based virtual screening (VS) in early-stage drug discovery has led to various successful leads (Damm-Ganamet et al, 2019), whereas, numerous problems of VS programs persist, including the inaccuracy of the scoring functions (SF) and the bias of the metrics used in the performance evaluation (Kellenberger et al., 2008; Scior et al., 2012), which promotes the application of more accurate and complex SFs and metrics borrowed from other fields. Correspondingly, the results emerging from our project have revealed that Vina, which has higher frequency of being employed in drug discovery projects and relatively more accurate SF, can still produce unsatisfactory results, while LIDAEUS, a program consisting of a simple posing algorithm and SF may outperform Vina in specific cases. More significantly, with simple optimizations, LIDAEUS can display similar level of performance as Vina. Additionally, the results have provided an approach to selecting suitable metrics employed in a VS performance evaluation and evaluated the applicability of these metrics addressing the real-world problems.

Considerations for VS performance evaluation

A VS performance evaluation may traditionally measure the scoring power and predictive power of the VS tools, while in our project, attention was only paid to the predictive power, the ability to enrich active molecules, which is due to the simplification we made to this evaluation, treating VS programs as binary classifiers. The performance of LDIAEUS was compared to the reference VS program, Vina. Although we assessed the search space impact on Vina's performance and attempted to run Vina with the optimal parameters, the stochastic algorithm embedded in Vina may still bring uncertainty to its performance. Additionally, the parameters derived from the assessment carried out on one target cannot guarantee the optimal global performance of Vina.

Bias of performance metrics

Figure 7 showed the correlations of AUC-ROC, EF and BEDROC, revealing the fact that they are intimately related to each other under some circumstances, which is in accord with the statement of Empereur-Mot et al. (2015) and assisted us to decide to use AUC-ROC and EF as the metrics evaluating the predictive power of a VS program. As for the bias of AUC-ROC and EF which is emphasized by Truchon and Christopher (2007) and Nichollos (2008) that their values are dependent on the ratio of the active compounds and the total compounds, however, in our study, such ratio is identical among different targets' ligand data sets, so this bias was eliminated and the reliability of the comparison of two VS programs against the same target was guaranteed.

"Early recognition" problem in VS performance evaluation

Although whether a VS program can rank active compounds very early in a large amount of compounds is crucial and numerous metrics was invented to address it, importance was not attached to such problem in our project considering the ambiguous "earliness" and the relatively small size of the ligand data set, which renders such issue less realistic. For instance, in the real-world projects of Dighe et al. (2016), initially, the top 10% of a screened result of 567,981 compounds were used in the next *in silico* screening phase, whereas, when the total compounds were reduced to 2,839, approximately 57% of the compounds were selected to inspect using another method. By contrast, in the project of Houston et al. (2015), about 12.5% of a ranked list of 2,120 molecules succeeded in entering manual selection phase. Above instances indicates that when the number of compounds falls into an acceptable range, which might be less than 3,000, researchers may tend to select molecules according to another criterion rather than the scores generated by the VS programs. Therefore, in our case, the "early recognition" problem might not be vital to our screening against data set of 1,240 molecules.

Grading AUC-ROC values

To compare the VS performance of LIDAEUS and Vina over 81 targets, the AUC-ROC values were grouped into 4 grades, namely, "**distinction**", "**merit**", "**pass**" and "**fail**", which were intensely used all through our project. However, such a classification method, deeming AUC-ROC over 0.70 as the optimal performance, may not be suitable for employing in the evaluation of VS programs other than LDIAEUS because it is based on a pre-evaluation of LIDAEUS where the occurrence of AUC-ROC over 0.70 is rare. But, given that a series of research (France et al., 2015; Healy et al., 2015; Houston et al., 2015) employed LIDAEUS in pre-screening large compound libraries in order to scale down to libraries to a suitable size for Vina screening, the above criterion for LIDAEUS's best performance seems to be reasonable and enough. Indeed, results (**Figure 8, 18 & 19**) indicate that with default settings, only a little portion of the results of LIDAEUS can produce AUC-ROC greater than 0.70, which might be the reason why researchers tended to use it in pre-screen for reducing libraries sizes. Based on above observation, subsequent improvements made to LIDAEUS were all evaluated by whether an increase of the number of results with AUC-ROC over 0.70 occurred.

From benchmarking data to mimetic real-world data

Our large-scale virtual screening experiment is a simulation employing LIDAEUS in a real-world lead identification project. LIDAEUS's unique advantage in screening against ER-beta was proven by the result (**Table 6**) where 9 out of 40 active compounds were present in the top 1,000 compounds. More surprisingly, 3 actives appeared in the first 40 molecules. With such impressive performance of LIDAEUS, the subsequent enrichment using Vina seemed to be less important. However, the success of this experiment might be attributed to the "artificial enrichment" bias. Molecules that have physicochemical similarities to the actives were eliminated from the data set, which might cause an accumulation of compounds that are hugely different from the active compounds, leading to an over-optimal screened

result. Indeed, no pre-screen was carried out for this data set to classify the molecules by their structures for avoidance of “artificial enrichment” bias. But, on the other hand, the only difference between this simulation and a real-world project is that the number of some active compounds is certain and the concept of “decoy bias” only exists in evaluating the quality of a benchmarking data set. Although we were conducting benchmarking, we were also simulating the process in reality. The significance of this experiment is that LIDAEUS can find the actives out of 54,066 compounds rather than why LIDAEUS can find the actives.

File format error

It is pointed out by Kirchmair et al. (2008) that the errors introduced by interconversion of file formats are normal and severe. As we reported in the results (**Table 3**), the file conversion programs in LIDAEUS can lead to the generation of disordered sitepoints, which could hardly be noticed in automated successive screenings against multiple targets. Besides, the abnormal format of LIDAEUS’s output file can cause misjudgements when manually inspecting in molecular graphic software. Since a VS program is, in essence, a product, above risks should also be evaluated and reported as a part of the VS performance assessment.

Considerations for the benchmarking data impacts

Cofactors in the target structures & Flexibility of target structures

As the limitation of LIDAEUS, every cofactor including metals were removed from the target structures, noted in **Table 2**. While the importance of these cofactors for protein-ligand binding were emphasised in the literatures associated with their PDB id, Vina and LIDAEUS can still generate results with satisfactory AUC-ROC in the absence of these crucial atoms or molecules. As noted in **Appendix 5,** in Vina’s screening against histone deacetylase 2 (**HDAC2**) which lost the vital zinc atom coordinated by Asp181, His183, and Asp269 after being processed (PDB id: 3MAX), an AUC-ROC of 0.89 was obtained from the result, whereas, in a similar case,

screening against glutaminyl-peptide cyclotransferase (**QPCT**) without the significant zinc atom, Vina failed to outperform random selection. Surprisingly, on the contrary, higher AUC-ROC was obtained from LDIAEUS's results before and after optimization. Similar phenomenon also occurred in the screenings against targets with high-flexibility binding sites (**Table 4**).

Above abnormal and fortuitous results may be related to what Scior et al. (2012) had stressed, a prevalent issue in VS program, assigning high scores to true actives with wrong predictive binding poses. However, screening against targets, heat shock protein 90 (**HSP90**) and SARS-HCoV, either with highly flexible binding site or without out key zinc atom, reminded us that such results of luck could not represent the performance of either Vina or LDIAEUS. In the docking experiment of HSP90 and SARS-HCoV, neither Vina nor LIDAEUS can perform better than a random selection regardless of optimization, indicating that the presence of cofactors is still vital to docking and, considering that Vina and LIDAEUS both are structure-based VS programs, the previous excellent result can just be attributed to luck and uncertain bias in the ligand data sets.

To address the cofactor interactions and receptor flexibility in VS, associated ligand-cofactor interaction can be added to LIDAEUS's sitepoints generation system and ensemble docking may be adopted, which utilizes different target conformers for docking as a compromising solution to target flexibility. Unfortunately, we failed to re-build a functioning system generating sitepoints in the presence of cofactors in LIDAEUS and considering the potential flexibility of each target, there was no additional time and computing resources for ensemble docking which needs the generation of each target's conformational snapshots using molecular dynamics (MD) simulation.

Multiple ligand conformers

For a more comprehensive comparison between the performance of LDAEUS and Vina, we initially utilized ligand conformers generated by

OpenBabel (O'Boyle et al., 2011) to implement a pseudo flexible docking in LIDAEUS, the result of which reveals an overall performance improvement in LIDAEUS (**Figure 8, B**). However, as Waszkowycz et al. (2011) suggested that suitable binding conformations should be more likely to be present in a considerably large conformer library, while due to limited computing resource, only 9 conformers were generated for each ligand in a data set, which lowered the probability of the presence of binding-favour conformers. Therefore, we subsequently used the conformers generated by Vina in the docking process as the input ligand data, which meant that the binding energy was taken into account. As noted in **Figure 8 (C)**, compared to the previous docking (**Figure 8, B**), with the same amount of results producing AUC-ROC over 0.70, more results fell into "**merit**" area where the related AUC-ROC should be over 0.60. Still, with regard to the performance of Vina (**Figure 8, D**), such improvement was too slight. Bauer et al. (2013) reported that decoys with latent activity could possibly embed in the ligand data set of RXR, which might result in LIDAEUS's unsatisfactory performance in screening against this target. Consequently, our efforts were dedicated toward improving LIDAEUS's performance by changing parameters to affect its posing mechanics and SF using RXR as a major example.

Considerations for the LIDAEUS's performance optimizations

Sitepoints variances

The result-lead optimization experiment (**Figure 12**) revealed the sitepoints generation parameters positive impact on the VS performance of LIDAEUS except the modification made to target SARS-CoV's sitepoints generation. However, this experiment also showed that the optimal parameter modifications were receptor-specific. A set of parameters leading to global optimal were not exhibited. But, by analysing the atom type of a set of sitepoints generated with extreme parameters in the screening against RXR (**Table 5**), the dominant factor of the sitepoints impact on the performance of LIDAEUS was revealed, that is, the distribution of the atoms in a set of sitepoints, providing the direction of making improvement to

LIDAEUS. Thus, either changing the number of sitepoint to affect the atom distribution or manually modifying the distribution, namely, re-shaping the sitepoints and fusing sets of sitepoints with different distributions, led to success in improving LIDAEUS's performance (**Figure 13, A, Figure 14 & Figure 15**). However, above approaches all suffered from instability and under some circumstances, they worsened the performance of LIDAEUS (**Figure 13, A; Figure 14, C; Figure 15, C**). A safer but less effective method is to increase the docking fitness tolerance (tol) to fit more compounds onto the sitepoints, which led to a LIDAEUS performance plateau after several tol increments with exponential increase in computational time (**Figure 13, B**). It should be noted that above experiments on the sitepoints distribution were only carried out on RXR and ERBB2, which means that exceptions may occur when employing these methods in other targets.

Entropy calculation in VS

Because of the extreme simplicity of the entropy calculation used in our project, a worse overall performance of LIDAEUS occurred, as noted in **Figure 18 (B)**. Such outcome was not beyond our expectation because the way we treated amino acid side chain conformational entropy and the lost water entropy is over-simple and, to some extent, wrong. Despite the side chain conformational entropy derived from experiments (Doig & Sternberg, 1995), treating every ligand-contacting side chain as flexible side chain ignored the constraints from the geometry of the binding site as well as the binding (Gaudreault et al., 2012). As for the consideration of water entropy, it is more problematic. The entropy calculation simply took in account a limited and small amount of water molecules in the binding process, those attached to molecules that can form hydrogen bonding, while, in fact, bulk solvent molecules, including water molecules, fill the cavities of proteins and surround apolar ligands (Breiten et al., 2013). Obviously, such calculation can by no means produce accurate entropic value and improve the SF of LIDAEUS.

Re-scoring

The built-in knowledge-based SF, PIP score, of LIDAEUS was used in re-scoring the poses of ligands generated in the process of rigid docking. **Figure 16** showed that the scale of resulting improvements was similar to the scale of improvements made by using multiple ligand conformers as input data. However, when combining the PIP scores with the original energy score calculated by LIDAEUS's SF, there was no occurrence of improvements but some setbacks (**Figure 17, A, B, C & D**). Additionally, a fully decline of LIDAEUS's performance occurred when considering both the PIP scores and the problematic entropic values in energy calculation (**Figure 17, E, F, G & H**). Notwithstanding above disappointing results, **Figure 18 (C, D, E & F)** revealed that re-parameterizing the energy calculation can bring impressive improvements to LIDAEUS's performance. In the pure enthalpy calculation, suitable weights of vDW, HBD and HBA was exhaustively explored for each target, seeking an optimal result producing the best AUC-ROC. Same procedure was performed in the combination of enthalpy and entropic terms, notably, the unweighted enthalpy, the conformational side chain entropy and the lost water entropy. Suitable PIP scores in combination of different energy scores for each target were also explored. Interestingly, with the addition of conditions, the overall performance of LIDAEUS became superior (**Figure 18, C, D, E & F)**. With the use of PIP scores combined with weighted enthalpy and entropy, LIDAEUS ultimately outperformed Vina (**Figure 18, F**). However, when we emphasized the EF 1% value of the results (**Figure 19, A**), the weights of each method for each target needed to modify. While Kolb et al. (2008) reported that a simple SF consisting of only vDW may have poorer predictive power than a relatively complete SF considering vDW and electrostatics has, our observation indicates the contrary. Regardless of either the objective or the kind of method, there was a common pattern of the weight's assignment, that is, one energy term accounting for a dominant weight and the others sharing the little ones. However, such SFs derived from re-parameterization might belong to target-biased SF, as Cheng et al. (2012) suggested. In order to obtain results with

higher accuracy, it is not enough to parameterize the SF against one structure of one target. Hence, our optimized energy calculation may merely suitable for such specific targets' structures in our project.

Considerations for binding site clustering

While we attempted to cluster targets' binding sites based on the chemical features and a geometric descriptor borrowed from small molecule shape recognition, **Figure 20 (A, B, C & D)** reveals that no strong correlations of LIDAEUS's performance and any set of targets can be observed. Besides, the justification of above clustering is doubtful. Initially, from the binding site extraction phase, problems might be already embedded, notably, the distance judging contacting amino acids, the choice of ligands and the ignorance of cofactors and binding site flexibility. The distance we selected, 3.5 Å, might not be enough for identification of al the contacting amino acids. Additionally, ligands from different structures were extracted to detect amino acid residues in the screening structure, which might introduce biases that some key contacting amino acids being neglected because the binding modes of each ligand are not identical, leading to different conformations of binding site. When a ligand was placed into a different structure, the relative position of the ligand and the amino acid had changed, thus some amino acids not being deemed as contacted. Cofactors are indeed significant components of binding sites but our atom-type based geometric descriptor failed to take the presence of cofactors into account. In addition, extracted binding sites, unlike small molecule with connected body, may have gap between key amino acids, thus whether this shape recognition approach is suitable to depict a binding site is in doubt. More crucially, the flexibility of binding sites can hugely affect the comparison between binding sites. As **Table 4** showed, three targets, HIV1RT, HSP90 and VEGFR2, possessing high-flexibility binding sites, may bring uncertainties into the comparison. To address this problem, a superior method is employing MD to explore the similarities between flexible binding sites.

Conclusion

In our project, we exhaustively explore the applicability of the prominent metrics and the performance of LIDAEUS in a VS evaluation. We succeed in making improvements on LIDAEUS's performance using targetbiased SF. However, due to the finite number of training structures, our improved SF should be lack of adaptivity when using other structures as input data. Nevertheless, we still provided a pragmatic guidance on employing LIDAEUS in research with its maximum performance.

Acknowledgements

I would like to express my gratitude to Dr. Paul Taylor for his patient guidance throughout this project, and his broad knowledge in virtual screening inspiring me to explore the full potential of LIDAEUS.

Appendix 1. PDB id for each target in this project.

Target	PDB id	Target	PDB id	Target	PDB id
11betaHSD1	3tfq	FKBP1A	2dg3	PDE5	1xp0
17betaHSD1	3klm	FXA	1f0r	PDK1	2xch
A2A	3eml	GBA	2wcg	PI3Kg	3dbs
ACE	1uze	GR	1nhz	PIM-1	3r04
ACE2	1r4l	GSK3B	3i4b	PIM-2	2iwi
ACHE	1eve	HDAC2	3max	PNP	1b8o
ADAM17	3ewj	HDAC8	3sff	PPARA	2p54
ADRB2	3ny9	HIV1PR	3nu3	PPARG	1fm9
AKT1	3qkl	HIV1RT	1s6p	PR	2w8y
ALR2	1ah3	HMGR	1hw8	PRKCQ	1xjd
AR	1e3g	HSP90	1uy6	PYGL-in	1xoi
AURKA	3fdn	IGF1R	3nw7	PYGL-out	3dds
AURKB	2vgo	INHA	1p44	QPCT	2afx
BCL2	2w3l	ITK	3mj1	ROCK-1	3v8s
BRAF	3skc	JAK3	3lxl	RXR	2p1t
CATL	3bc3	JNK1	3elj	SARS-HCoV	2z94
CDK2	1ckp	JNK2	3npc	SIRT2	5yqo
COX1	3kk6	JNK3	2b1p	SRC	2src
COX2	1cx2	KIF11	3k5e	Thrombin	3rm2
CTSK	3kx1	LCK	3mpm	TIE2	2oo8
CYP2A6	1z11	MDM2	3lbk	TK	1w4r
DHFR	1s3v	MK2	3kc3	TP	1uou
EGFR	1m17	MMP2	1hov	TPA	1a5h
EPHB4	2vwz	NA	1a4g	TS	1i00

ER-beta	3oll	P38-alpha	1ouk	uPA	3mhw
ERBB2	3pp0	PARP-1	3l3m	VEGFR1	3hng
FGFR1	1agw	PDE4B	3frg	VEGFR2	3c7q

Appendix 2. Amino acid classification

Amino acid	Hydrophobicity	Hydrogen donor atoms	Hydrogen acceptor atoms	Side chain classification
ALA	0.31	None	None	Hydrophobic aliphatic
ARG	-1.01	NE, NH1 (2), NH2 (2)	None	Positive charged
ASN	-0.60	ND2 (2)	OD1 (2)	Polar uncharged
ASP	-0.77		OD1 (2), OD2 (2)	Negative charged
CYS	1.54	None	None	Hydrophobic aliphatic
GLN	-0.22	NE2 (2)	OE1 (2)	Polar uncharged
GLU	-0.64		OE1 (2), OE2 (2)	Negative charged
HIS	0.13	ND1, NE2	ND1, NE2	Positive charged
ILE	1.80	None	None	Hydrophobic aliphatic
LEU	1.70	None	None	Hydrophobic aliphatic
LYS	-0.99	NZ (3)	None	Positive charged
MET	1.23	None	None	Hydrophobic aliphatic
PHE	1.79	None	None	Hydrophobic aromatic
PRO	0.72	None	None	Hydrophobic aliphatic
SER	-0.04	OG	OG (2)	Polar uncharged
THR	0.26	OG1	OG1 (2)	Polar uncharged

TRP	2.25	NE1	None	Hydrophobic aromatic
TYR	0.96	OH	OH	Hydrophobic aromatic
VAL	1.22	None	None	Hydrophobic aliphatic
GLY	0	None	None	Hydrophobic aliphatic

Note: the number in bracket denotes number of "sp hydrogens" that a donor atom can donate or the number of hydrogen bonds that an acceptor atom can accept, if they are more than one.

Appendix 3. Processing on target structures

Target	PDB code	Processing	Ligand	Metal	Cofactor
11betaHSD1	3tfq	Keep chain B; Discard cofactor NAP	07M		NAP
17betaHSD1	3klm	Discard GOL	DHT (2 orientations)		
A2A	3eml	Discard STE and SO4	ZMA		
ACE	1uze	Discard Cl and disordered GLY	EAL	Zn	
ACE2	1r4	Discard disordered chain B C D E; discard NAG; discard Cl; remove incomplete GLN-618	XX5	Zn	
ACHE	1eve	Discard NAG	E20		
ADAM17	3ewj	Keep chain A; discard Zn	INN	Zn	
ADRB2	3ny9	Discard CLR and OLC	JSZ		
AKT1	3qkl	Discard chain C; discard modified residue TPO	SMR		
ALR2	1ah3	Discard cofactor NAP; remove modified residue AYA	TOL		NAP

AR	1e3g		R18		
AURKA	3fdn		MMH		
AURKB	2vgo	Discard chain A C D; remove modified residue TPO	AD5		
BCL2	2w3l	Discard chain B	DRO		
BRAF	3skc	Discard chain B	BR2		
CATL	3bc3	Discard chain B; remove modified residue CSD	OPT		CSD
CDK2	1ckp	Discard EDO	PVB		
COX1	3kk6	Discard chain B (not completely occupied by the ligand); discard NAG, NDG, MAN, BMA, BOG and HEM	CEL		
COX2	1cx2	Discard chain B; discard NAG and HEM	s58		
CTSK	3kx1	Discard SO4	KX1		
CYP2A6	1z11	Keep chain A; discard HEM	8MO		HEM
DHFR	1s3v	Discard SO4	TQD		
EGFR	1m17		AQ4		
EPHB4	2vwz	Discard Mg	7X6		
ER-beta	3oll	Discard chain B C D; remove modified residue PTR	EST		
ERBB2	3pp0	Discard chain A	03Q		
FGFR1	1agw	Discard chain B	SU2		
FKBP1A	2dg3	Discard GOL	RAP		
FXA	1f0r	Discard chain B and Ca	815		
GBA	2wcg	Discard chain A; discard SO4, Cl, FUC, BMA and NAG	MT5		
GR	1nhz	Discard HEZ	486		
GSK3B	3i4b	Discard chain B	Z48		
HDAC2	3max	Discard chain B and C; Discard	LLX	Zn	

		Ca, Na, NHE and ZN			
HDAC8	3sff	Discard K; discard Zn	ODI	Zn	
HIV1PR	3nu3	Discard Cl and GOL; discard chain A	486		
HIV1RT	1s6p	Discard chain B; discard Mg	IET		
HMGR	1hw8	Discard chain A C D; discard ADP	114		
HSP90	1uy6	remove incomplete LYS	PU3		
IGF1R	3nw7		LGV		
INHA	1p44	Discard chain B C D E F and cofactor NAD	GEQ		NAD
ITK	3mj1		614		
JAK3	3lxl		IZA		
JNK1	3elj		GS7		
JNK2	3npc	Discard chain BME	B96		
JNK3	2b1p	Discard SO4	AIZ		
KIF11	3k5e	Discard chain B; discard ADP and Mg	K5E	Mg	
LCK	3mpm	Discard EDO	5LK		
MDM2	3lbk	Discard SO4	K23		
MK2	3kc3	Discard chain B C D E F G H I J K L	MK2		
MMP2	1hov	Discard Ca and Zn	I52	Zn	
NA	1a4g	Discard chain B; discard NAG and Ca	ZMR		
P38-alpha	1ouk	Discard SO4	084		
PARP-1	3l3m		A92		
PDE4B	3frg	Discard GOL, ARS, Zn and Mg	SK4	Zn and Mg	
PDE5	1xp0	Discard Zn and Mg	VDN	Zn and Mg	
PDK1	2xch	Discard GOL and SO4; remove	CKG		

		modified residue SEP			
PI3Kg	3dbs		GD9		
PIM-1	3r04	Discard IMD	UNQ		
PIM-2	2iwi	Discard chain A	HB1		
PNP	1b8o	Discard Mg, PO4	IMH		
PPARA	2p54	Discard chain B	735		
PPARG	1fm9	Keep chain D	570		
PR	2w8y	Discard chain B; discard SO4; remove incomplete LYS 933	486		
PRKCQ	1xjd	Remove modified residues TPO and SEP	STU		
PYGL-in	1xoi	Discard PLP and NBG	two molecules of 288		
PYGL-out	3dds	Discard NBG, PO4, PLP, CFF and MPD; remove modified residue SEP	26B		
QPCT	2afx	Discard chain B; discard SO4 and Zn	1BN	Zn	
ROCK-1	3v8s	Discard chain A C D;	0HD		
RXR	2p1t	Discard chain B	3TN		
SARS-HCoV	2z94	Removed Zn	TLD	Zn	
SIRT2	5yqo		L5C		
SRC	2src	Remove modified residue PTR	ANP		
Thrombin	3rm2	Discard chain L I; discard PO4, GOL, Na and NAG	S00		
TIE2	2oo8		RAJ		
TK	1w4r	Discard chain B C D E F G H	TTP		
TP	1uou		CMU		
TPA	1a5h	Discard chain C D B	BBA		
TS	1i00	Discard chain B	D16		

uPA	3mhw	Discard SO4 and cofactor UMP	ABV		UMP
VEGFR1	3hng	Discard Cl	8ST		
VEGFR2	3c7q	Discard SO4; remove modified residue CME, PTR	XIN		

Appendix 4. Target classification

Targets	Hydrophobic side chain	Polar side chain	Without HBD/HBA atoms	With HBD/HBA atoms	Hydrophobicity
11betaHSD1	78.95%	21.05%	52.00%	48.00%	16.62
17betaHSD1	60.00%	40.00%	42.86%	57.14%	3.73
A2A	66.67%	33.33%	46.67%	53.33%	12.42
ACE	33.33%	66.67%	10.53%	89.47%	0.26
ACE2	27.27%	72.73%	5.88%	94.12%	0.23
ACHE	75.00%	25.00%	20.00%	80.00%	10.78
ADAM17	53.33%	46.67%	33.33%	66.67%	5.39
ADRB2	57.14%	42.86%	9.09%	90.91%	3.99
AKT1	50.00%	50.00%	45.45%	54.55%	0.86
ALR2	73.68%	26.32%	36.00%	64.00%	20.12
AR	72.22%	27.78%	54.55%	45.45%	16.14
AURKA	82.35%	17.65%	68.42%	31.58%	12.27
AURKB	40.00%	60.00%	40.00%	60.00%	-0.17
BCL2	80.00%	20.00%	50.00%	50.00%	3.52
BRAF	60.00%	40.00%	41.67%	58.33%	7.57
CATL	75.00%	25.00%	57.14%	42.86%	4.47
CDK2	50.00%	50.00%	40.00%	60.00%	7.45
COX1	73.91%	26.09%	41.94%	58.06%	21.87
COX2	64.29%	35.71%	36.84%	63.16%	8.02
CTSK	50.00%	50.00%	33.33%	66.67%	0.85
CYP2A6	83.33%	16.67%	71.43%	28.57%	5.99
DHFR	50.00%	50.00%	38.10%	61.90%	7.89
EGFR	70.00%	30.00%	60.87%	39.13%	16.32
EPHB4	55.56%	44.44%	50.00%	50.00%	5.18
ERBB2	61.54%	38.46%	42.42%	57.58%	15.73
ER-beta	25.00%	75.00%	20.00%	80.00%	-0.29
FGFR1	62.50%	37.50%	47.37%	52.63%	4.97
FKBP1A	71.43%	28.57%	38.89%	61.11%	14.66
FXA	66.67%	33.33%	42.86%	57.14%	3.36
GBA	37.50%	62.50%	10.00%	90.00%	3.6

GR	40.00%	60.00%	28.57%	71.43%	2.04
GSK3B	60.00%	40.00%	54.55%	45.45%	3.53
HDAC2	66.67%	33.33%	40.00%	60.00%	7.28
HDAC8	58.33%	41.67%	31.25%	68.75%	3.85
HIV1PR	62.50%	37.50%	62.50%	37.50%	0.71
HIV1RT	80.00%	20.00%	38.46%	61.54%	8.53
HMGR	31.58%	68.42%	24.00%	76.00%	-2.25
HSP90	55.56%	44.44%	34.78%	65.22%	7.71
IGF1R	53.85%	46.15%	46.67%	53.33%	2.7
INHA	90.91%	9.09%	75.00%	25.00%	10.74
ITK	66.67%	33.33%	61.54%	38.46%	7.7
JAK3	80.00%	20.00%	58.33%	41.67%	4.96
JNK1	61.90%	38.10%	50.00%	50.00%	9.51
JNK2	63.64%	36.36%	58.33%	41.67%	6.48
JNK3	60.71%	39.29%	51.52%	48.48%	9.53
KIF11	33.33%	66.67%	33.33%	66.67%	-2.03
LCK	60.00%	40.00%	41.67%	58.33%	4.34
MDM2	80.00%	20.00%	66.67%	33.33%	4.85
MK2	57.14%	42.86%	50.00%	50.00%	9.42
MMP2	65.22%	34.78%	46.67%	53.33%	17.1
NA	25.00%	75.00%	7.69%	92.31%	-4.22
P38-alpha	66.67%	33.33%	50.00%	50.00%	8.95
PARP-1	44.44%	55.56%	14.29%	85.71%	-0.35
PDE4B	53.33%	46.67%	21.74%	78.26%	11.5
PDE5	50.00%	50.00%	33.33%	66.67%	1.57
PDK1	47.06%	52.94%	35.00%	65.00%	2.46
PI3Kg	52.38%	47.62%	37.50%	62.50%	8.24
PIM-1	53.85%	46.15%	50.00%	50.00%	4.11
PIM-2	62.50%	37.50%	62.50%	37.50%	4.09
PNP	55.56%	44.44%	30.77%	69.23%	2.74
PPARA	76.00%	24.00%	53.13%	46.88%	28.26
PPARG	75.86%	24.14%	55.56%	44.44%	29.77
PR	66.67%	33.33%	45.45%	54.55%	8.38
PRKCQ	54.55%	45.45%	41.67%	58.33%	2.21
PYGL-in	37.50%	62.50%	9.09%	90.91%	1.68
PYGL-out	44.44%	55.56%	40.00%	60.00%	0.76
QPCT	25.00%	75.00%	20.00%	80.00%	0.1
ROCK-1	66.67%	33.33%	61.54%	38.46%	6.28
RXR	73.33%	26.67%	61.11%	38.89%	13.15
SARS-HCoV	0.00%	0.00%	0.00%	0.00%	0
SIRT2	72.41%	27.59%	57.14%	42.86%	27.59
SRC	43.75%	56.25%	28.57%	71.43%	2.21
Thrombin	62.50%	37.50%	28.57%	71.43%	7.01

TIE2	80.00%	20.00%	64.71%	35.29%	14.69
TK	57.89%	42.11%	43.48%	56.52%	9.63
TP	22.22%	77.78%	6.67%	93.33%	-0.47
TPA	61.54%	38.46%	35.29%	64.71%	4
TS	85.71%	14.29%	50.00%	50.00%	7.72
uPA	62.50%	37.50%	42.86%	57.14%	9.87
VEGFR1	50.00%	50.00%	50.00%	50.00%	2.96
VEGFR2	63.16%	36.84%	60.00%	40.00%	12.68

Appendix 5. AUC-ROC derived from different methods for each target

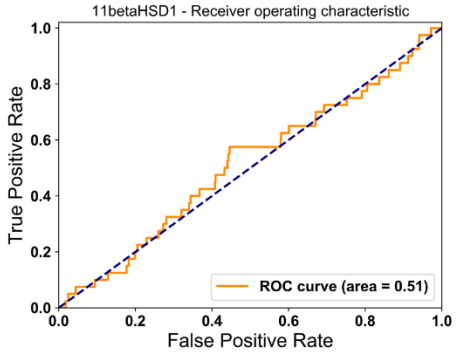
>0.70 | >0.60 | >0.50 | <0.50

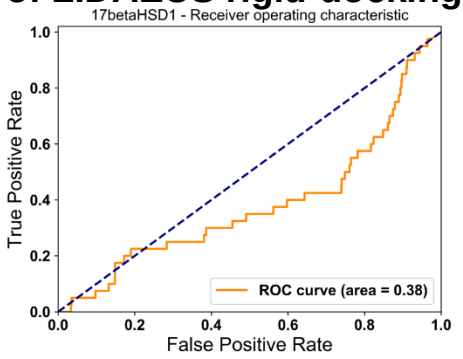
Note: red cell with value equal to 0.50 denotes pseudo random selection.

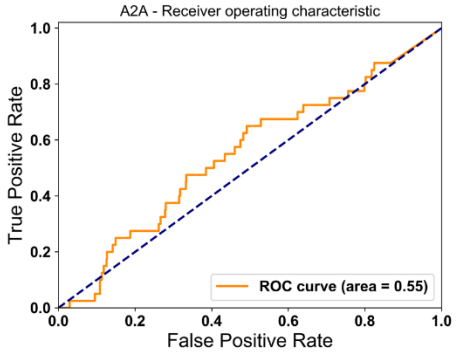
Target	Rigid docking	Flexible docking (conformers generated by openbabel)	Flexible docking (conformers generated by Vina)	Enthalpy + Entropy	PIP_1	PIP_2	PIP_3	PIP_4	PIP_1 * unweighted Enthalpy	PIP_2 * unweighted Enthalpy	PIP_3 * unweighted Enthalpy	PIP_4 * unweighted Enthalpy	PIP_1 * unweighted (Enthalpy + Entropy)	PIP_2 * unweighted (Enthalpy + Entropy)	PIP_3 * unweighted (Enthalpy + Entropy)	PIP_4 * unweighted (Enthalpy + Entropy)	Weighted Enthalpy	Weighted (Entahlpy + Entropy)	PIP * weighted Enthalpy	PIP * weighted (Enthalpy + Entropy)	Vina
11betaHSD1	0.51	0.52	0.51	0.51	0.53	0.56	0.50	0.49	0.50	0.49	0.51	0.49	0.51	0.49	0.51	0.50	0.54	0.54	0.55	0.55	0.59
17betaHSD1	0.38	0.35	0.39	0.36	0.37	0.39	0.45	0.44	0.38	0.38	0.38	0.39	0.39	0.38	0.39	0.38	0.50	0.50	0.50	0.61	0.53
A2A	0.55	0.49	0.47	0.56	0.49	0.47	0.48	0.50	0.57	0.59	0.57	0.59	0.57	0.59	0.57	0.59	0.57	0.61	0.60	0.61	0.52
ACE	0.58	0.49	0.61	0.56	0.69	0.67	0.57	0.62	0.59	0.59	0.59	0.59	0.62	0.62	0.63	0.63	0.69	0.66	0.69	0.72	0.46
ACE2	0.49	0.45	0.45	0.46	0.72	0.61	0.66	0.53	0.56	0.56	0.58	0.56	0.62	0.60	0.64	0.60	0.63	0.75	0.61	0.75	0.52
ACHE	0.64	0.64	0.61	0.64	0.62	0.56	0.59	0.55	0.64	0.65	0.64	0.65	0.63	0.64	0.64	0.65	0.65	0.65	0.68	0.67	0.70
ADAM17	0.42	0.50	0.60	0.43	0.59	0.57	0.62	0.58	0.42	0.42	0.43	0.42	0.41	0.41	0.42	0.41	0.61	0.60	0.61	0.66	0.53
ADRB2	0.49	0.51	0.60	0.49	0.38	0.41	0.38	0.42	0.52	0.53	0.51	0.53	0.52	0.52	0.51	0.52	0.50	0.53	0.54	0.62	0.66
AKT1	0.55	0.57	0.52	0.55	0.57	0.60	0.60	0.62	0.54	0.58	0.54	0.59	0.54	0.58	0.54	0.58	0.59	0.69	0.62	0.63	0.68
ALR2	0.57	0.62	0.62	0.57	0.49	0.53	0.41	0.49	0.55	0.56	0.55	0.56	0.53	0.53	0.53	0.53	0.62	0.59	0.62	0.56	0.61
AR	0.58	0.52	0.49	0.58	0.60	0.57	0.53	0.55	0.57	0.54	0.57	0.55	0.56	0.53	0.57	0.54	0.62	0.59	0.62	0.58	0.65
AURKA	0.57	0.43	0.48	0.55	0.52	0.65	0.53	0.62	0.58	0.60	0.58	0.60	0.57	0.56	0.57	0.56	0.61	0.62	0.66	0.65	0.67
AURKB	0.38	0.45	0.34	0.40	0.41	0.48	0.44	0.45	0.39	0.37	0.39	0.36	0.36	0.34	0.36	0.33	0.50	0.52	0.50	0.52	0.64
BCL2	0.44	0.48	0.49	0.43	0.52	0.42	0.62	0.48	0.43	0.43	0.44	0.42	0.46	0.47	0.47	0.47	0.61	0.76	0.64	0.72	0.69
BRAF	0.46	0.37	0.59	0.45	0.40	0.50	0.47	0.52	0.45	0.44	0.45	0.45	0.46	0.45	0.46	0.47	0.63	0.53	0.65	0.64	0.68
CATL	0.59	0.54	0.51	0.58	0.54	0.40	0.51	0.41	0.58	0.58	0.59	0.58	0.58	0.57	0.59	0.58	0.70	0.65	0.71	0.63	0.67
CDK2	0.62	0.60	0.52	0.62	0.49	0.52	0.52	0.58	0.61	0.59	0.60	0.60	0.62	0.61	0.61	0.62	0.65	0.64	0.64	0.65	0.60
COX1	0.68	0.59	0.60	0.65	0.78	0.78	0.66	0.75	0.70	0.69	0.70	0.69	0.67	0.67	0.67	0.67	0.73	0.68	0.75	0.75	0.70
COX2	0.58	0.51	0.46	0.51	0.71	0.77	0.72	0.77	0.61	0.62	0.62	0.62	0.60	0.63	0.60	0.62	0.74	0.58	0.78	0.79	0.88
CTSK	0.51	0.57	0.58	0.51	0.53	0.49	0.70	0.66	0.47	0.51	0.50	0.54	0.44	0.44	0.44	0.44	0.66	0.62	0.68	0.66	0.66
CYP2A6	0.56	0.46	0.46	0.47	0.68	0.68	0.43	0.49	0.58	0.58	0.55	0.57	0.64	0.65	0.64	0.65	0.61	0.72	0.63	0.70	0.39
DHFR	0.47	0.41	0.46	0.47	0.43	0.49	0.47	0.51	0.48	0.46	0.45	0.45	0.48	0.45	0.45	0.45	0.50	0.50	0.50	0.50	0.43
EGFR	0.61	0.63	0.61	0.60	0.55	0.61	0.51	0.60	0.64	0.64	0.64	0.64	0.63	0.62	0.63	0.62	0.63	0.61	0.66	0.65	0.59
EPHB4	0.40	0.62	0.49	0.40	0.55	0.59	0.52	0.51	0.40	0.41	0.40	0.39	0.41	0.41	0.40	0.39	0.58	0.50	0.63	0.56	0.77
ERBB2	0.75	0.57	0.58	0.75	0.61	0.65	0.62	0.63	0.71	0.70	0.72	0.70	0.65	0.63	0.65	0.64	0.76	0.78	0.78	0.86	0.74
ER-beta	0.56	0.54	0.54	0.55	0.61	0.68	0.64	0.67	0.58	0.63	0.58	0.63	0.59	0.65	0.59	0.65	0.61	0.58	0.71	0.66	0.63
FGFR1	0.57	0.61	0.57	0.61	0.53	0.54	0.64	0.63	0.55	0.54	0.56	0.54	0.50	0.51	0.49	0.50	0.68	0.71	0.65	0.69	0.72
FKBP1A	0.50	0.50	0.65	0.32	0.40	0.41	0.49	0.44	0.45	0.47	0.45	0.47	0.67	0.70	0.67	0.69	0.60	0.54	0.59	0.72	0.45
FXA	0.62	0.50	0.63	0.65	0.63	0.61	0.58	0.61	0.62	0.58	0.61	0.58	0.53	0.52	0.52	0.51	0.66	0.75	0.64	0.61	0.79
GBA	0.57	0.56	0.56	0.58	0.68	0.69	0.72	0.66	0.57	0.59	0.56	0.59	0.54	0.56	0.54	0.56	0.60	0.66	0.64	0.75	0.41
GR	0.64	0.66	0.74	0.64	0.80	0.73	0.76	0.68	0.61	0.57	0.61	0.59	0.62	0.59	0.62	0.59	0.75	0.64	0.78	0.83	0.44
GSK3B	0.45	0.52	0.46	0.45	0.46	0.47	0.51	0.45	0.43	0.45	0.44	0.45	0.42	0.43	0.43	0.43	0.50	0.50	0.52	0.51	0.62
HDAC2	0.54	0.63	0.67	0.52	0.46	0.42	0.43	0.44	0.53	0.53	0.53	0.53	0.54	0.54	0.54	0.54	0.58	0.62	0.57	0.59	0.89
HDAC8	0.43	0.51	0.52	0.43	0.51	0.53	0.56	0.59	0.44	0.43	0.44	0.43	0.45	0.44	0.45	0.44	0.54	0.56	0.54	0.62	0.74
HIV1PR	0.43	0.46	0.71	0.33	0.53	0.50	0.52	0.54	0.48	0.47	0.47	0.47	0.64	0.60	0.62	0.59	0.51	0.69	0.56	0.76	0.65
HIV1RT	0.62	0.61	0.56	0.60	0.63	0.66	0.56	0.66	0.64	0.64	0.63	0.63	0.62	0.62	0.61	0.62	0.62	0.62	0.64	0.65	0.47
HMGR	0.52	0.53	0.51	0.51	0.56	0.51	0.59	0.60	0.56	0.63	0.57	0.64	0.56	0.64	0.57	0.64	0.62	0.71	0.67	0.74	0.45
HSP90	0.47	0.47	0.51	0.48	0.44	0.47	0.37	0.50	0.48	0.41	0.49	0.41	0.48	0.41	0.50	0.41	0.54	0.58	0.54	0.50	0.43
IGF1R	0.54	0.32	0.40	0.55	0.47	0.53	0.54	0.53	0.54	0.52	0.55	0.53	0.54	0.51	0.54	0.53	0.58	0.70	0.57	0.58	0.80
INHA	0.64	0.64	0.61	0.61	0.59	0.65	0.53	0.71	0.63	0.63	0.62	0.62	0.62	0.63	0.62	0.63	0.71	0.69	0.72	0.67	0.52
ITK	0.64	0.59	0.51	0.64	0.67	0.69	0.71	0.72	0.59	0.56	0.60	0.57	0.58	0.53	0.59	0.55	0.67	0.66	0.66	0.71	0.53
JAK3	0.47	0.58	0.47	0.47	0.46	0.52	0.63	0.54	0.44	0.43	0.43	0.43	0.44	0.43	0.43	0.42	0.56	0.60	0.55	0.63	0.74
JNK1	0.55	0.49	0.54	0.62	0.60	0.67	0.52	0.65	0.56	0.56	0.55	0.56	0.48	0.48	0.48	0.48	0.58	0.71	0.60	0.72	0.65
JNK2	0.52	0.56	0.51	0.52	0.51	0.47	0.50	0.51	0.50	0.51	0.50	0.51	0.49	0.50	0.49	0.49	0.57	0.53	0.58	0.53	0.65
JNK3	0.61	0.57	0.60	0.65	0.86	0.86	0.74	0.82	0.66	0.64	0.66	0.64	0.49	0.49	0.49	0.49	0.71	0.76	0.74	0.81	0.71
KIF11	0.65	0.73	0.80	0.61	0.48	0.43	0.38	0.44	0.64	0.65	0.65	0.66	0.71	0.73	0.72	0.73	0.67	0.66	0.70	0.75	0.86
LCK	0.51	0.52	0.53	0.52	0.60	0.58	0.54	0.60	0.54	0.53	0.54	0.53	0.57	0.56	0.57	0.56	0.53	0.60	0.56	0.59	0.50
MDM2	0.61	0.62	0.68	0.54	0.60	0.66	0.71	0.49	0.59	0.59	0.59	0.59	0.64	0.64	0.64	0.64	0.62	0.61	0.68	0.83	0.45
MK2	0.69	0.64	0.57	0.69	0.69	0.77	0.69	0.75	0.66	0.66	0.67	0.66	0.64	0.64	0.65	0.64	0.71	0.75	0.68	0.71	0.85
MMP2	0.58	0.60	0.67	0.57	0.64	0.62	0.62	0.63	0.61	0.61	0.61	0.61	0.65	0.63	0.65	0.63	0.62	0.66	0.64	0.69	0.53
NA	0.67	0.58	0.61	0.68	0.76	0.72	0.65	0.64	0.64	0.68	0.64	0.69	0.62	0.68	0.63	0.69	0.80	0.73	0.81	0.73	0.55
P38-alpha	0.58	0.49	0.46	0.58	0.56	0.57	0.56	0.58	0.57	0.58	0.57	0.58	0.57	0.58	0.57	0.58	0.60	0.59	0.59	0.62	0.69
PARP-1	0.73	0.75	0.76	0.68	0.65	0.63	0.69	0.66	0.72	0.70	0.74	0.72	0.62	0.61	0.62	0.61	0.79	0.73	0.77	0.74	0.80
PDE4B	0.68	0.55	0.67	0.62	0.69	0.67	0.58	0.57	0.67	0.68	0.66	0.67	0.66	0.63	0.66	0.63	0.70	0.73	0.71	0.72	0.67
PDE5	0.54	0.42	0.47	0.54	0.46	0.53	0.57	0.58	0.61	0.59	0.61	0.59	0.63	0.60	0.62	0.60	0.58	0.66	0.62	0.64	0.61
PDK1	0.63	0.60	0.57	0.62	0.46	0.47	0.49	0.51	0.64	0.64	0.64	0.63	0.63	0.63	0.63	0.62	0.65	0.63	0.65	0.66	0.63
PI3Kg	0.49	0.51	0.53	0.49	0.60	0.63	0.63	0.56	0.49	0.47	0.49	0.47	0.49	0.46	0.49	0.47	0.55	0.55	0.57	0.63	0.72
PIM-1	0.56	0.71	0.65	0.56	0.63	0.61	0.56	0.60	0.56	0.55	0.56	0.55	0.54	0.53	0.53	0.53	0.61	0.58	0.60	0.62	0.69
PIM-2	0.60	0.72	0.63	0.59	0.49	0.51	0.51	0.50	0.59	0.59	0.59	0.58	0.55	0.55	0.56	0.55	0.61	0.60	0.60	0.59	0.64
PNP	0.66	0.74	0.74	0.65	0.62	0.68	0.58	0.67	0.67	0.65	0.66	0.66	0.63	0.61	0.64	0.61	0.68	0.67	0.68	0.68	0.74
PPARA	0.52	0.58	0.64	0.51	0.65	0.59	0.63	0.65	0.54	0.53	0.54	0.53	0.54	0.54	0.54	0.54	0.56	0.62	0.60	0.80	0.72
PPARG	0.47	0.47	0.52	0.47	0.54	0.58	0.55	0.57	0.59	0.56	0.58	0.57	0.58	0.57	0.58	0.58	0.51	0.69	0.60	0.72	0.66
PR	0.66	0.71	0.67	0.68	0.74	0.69	0.71	0.71	0.67	0.67	0.67	0.67	0.66	0.66	0.66	0.66	0.73	0.71	0.76	0.72	0.59
PRKCQ	0.57	0.52	0.48	0.57	0.54	0.58	0.56	0.63	0.59	0.56	0.59	0.54	0.59	0.55	0.59	0.54	0.60	0.80	0.62	0.62	0.71
PYGL-in	0.72	0.77	0.79	0.62	0.65	0.67	0.56	0.63	0.73	0.74	0.73	0.74	0.72	0.72	0.72	0.72	0.73	0.74	0.75	0.75	0.52
PYGL-out	0.47	0.55	0.54	0.46	0.54	0.56	0.54	0.62	0.44	0.46	0.46	0.47	0.45	0.47	0.47	0.48	0.51	0.52	0.52	0.67	0.81
QPCT	0.58	0.49	0.51	0.60	0.47	0.65	0.58	0.72	0.57	0.58	0.58	0.59	0.52	0.48	0.52	0.49	0.78	0.64	0.77	0.76	0.38
ROCK-1	0.45	0.47	0.42	0.45	0.54	0.53	0.61	0.58	0.45	0.45	0.44	0.44	0.45	0.46	0.44	0.44	0.50	0.59	0.50	0.57	0.52
RXR	0.30	0.25	0.19	0.28	0.44	0.49	0.51	0.48	0.27	0.26	0.27	0.26	0.30	0.28	0.29	0.28	0.52	0.50	0.55	0.63	0.69
SARS-HCoV	0.30	0.38	0.38	0.30	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.40
SIRT2	0.65	0.53	0.60	0.64	0.61	0.67	0.64	0.61	0.65	0.64	0.65	0.64	0.66	0.63	0.66	0.63	0.67	0.70	0.67	0.67	0.62
SRC	0.46	0.52	0.52	0.46	0.45	0.45	0.45	0.45	0.49	0.48	0.48	0.46	0.49	0.48	0.47	0.46	0.50	0.56	0.51	0.53	0.57
Thrombin	0.62	0.55	0.67	0.63	0.54	0.52	0.49	0.54	0.60	0.59	0.60	0.58	0.59	0.57	0.58	0.56	0.65	0.67	0.66	0.64	0.79
TIE2	0.45	0.43	0.43	0.42	0.35	0.36	0.35	0.41	0.42	0.43	0.43	0.42	0.47	0.48	0.47	0.48	0.54	0.50	0.50	0.52	0.63
TK	0.61	0.50	0.72	0.58	0.55	0.62	0.57	0.63	0.59	0.58	0.59	0.58	0.60	0.59	0.59	0.59	0.76	0.67	0.77	0.66	0.53
TP	0.53	0.57	0.57	0.52	0.48	0.49	0.45	0.47	0.52	0.53	0.52	0.52	0.51	0.52	0.52	0.52	0.54	0.54	0.55	0.53	0.45
TPA	0.74	0.71	0.75	0.74	0.68	0.66	0.58	0.63	0.72	0.75	0.71	0.74	0.71	0.72	0.71	0.71	0.76	0.85	0.79	0.76	0.75
TS	0.52	0.55	0.55	0.50	0.44	0.42	0.47	0.40	0.50	0.49	0.49	0.49	0.51	0.52	0.51	0.51	0.58	0.56	0.51	0.57	0.57
uPA	0.42	0.45	0.50	0.42	0.35	0.41	0.36	0.45	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.50	0.50	0.50	0.52	0.57
VEGFR1	0.57	0.59	0.61	0.57	0.48	0.51	0.45	0.55	0.57	0.57	0.57	0.58	0.57	0.56	0.57	0.58	0.60	0.62	0.62	0.58	0.67
VEGFR2	0.37	0.41	0.41	0.38	0.68	0.62	0.62	0.62	0.36	0.38	0.36	0.38	0.36	0.39	0.36	0.39	0.50	0.50	0.54	0.65	0.62

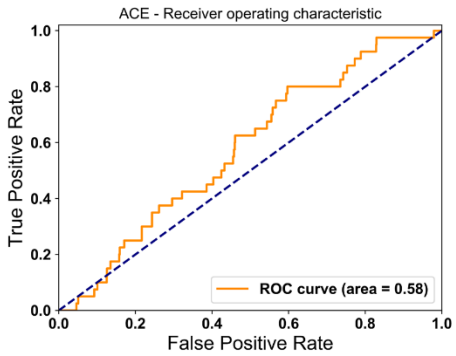
Appendix 6. ROC curves

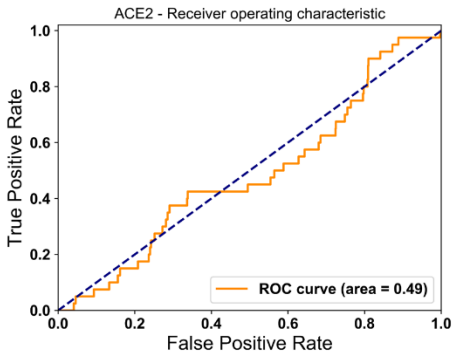
ROC of LIDAEUS rigid-docking results

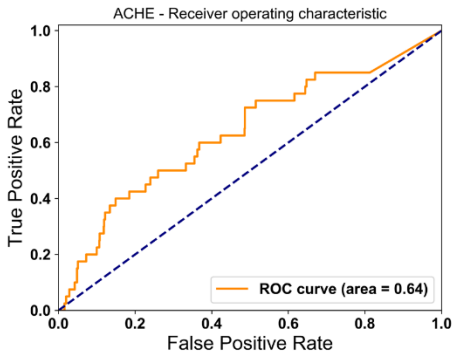


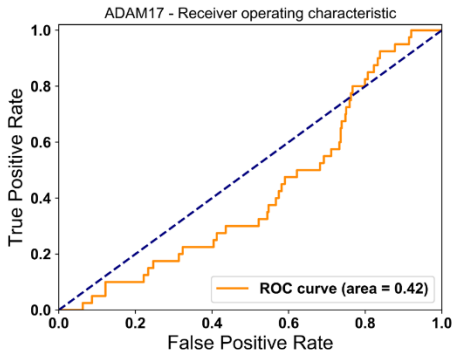


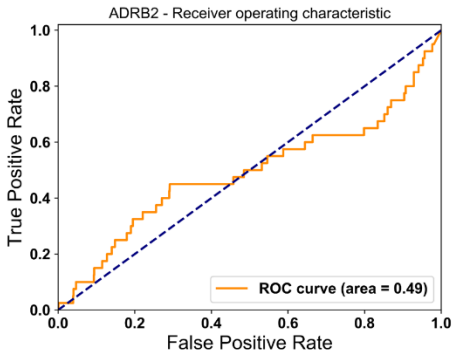


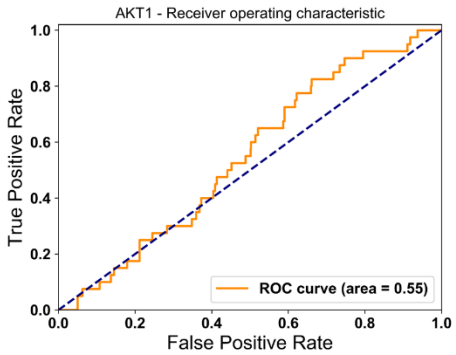


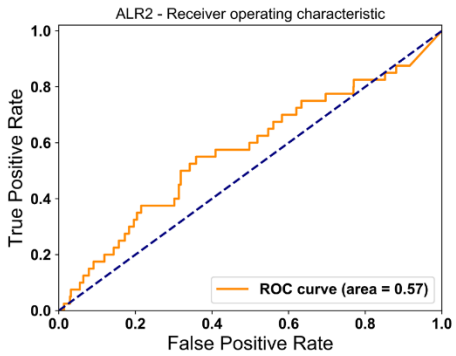


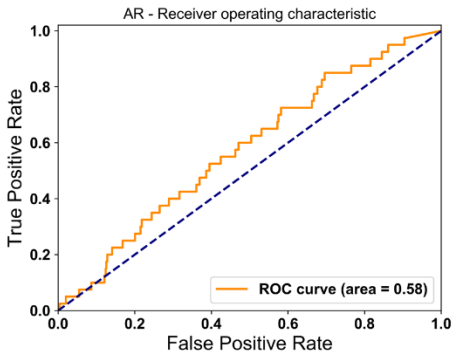


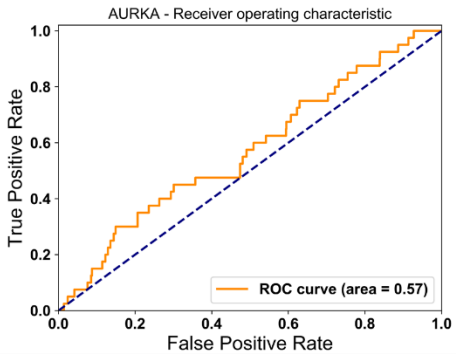


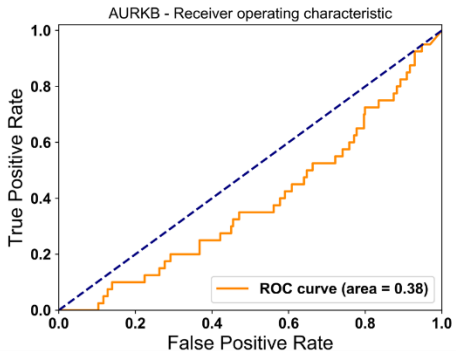


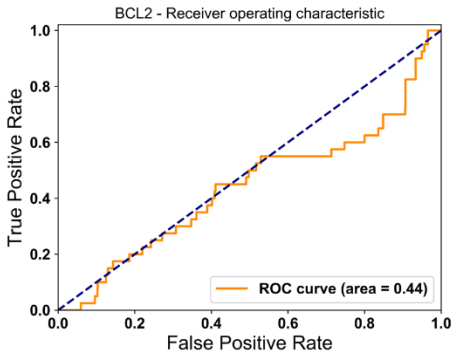


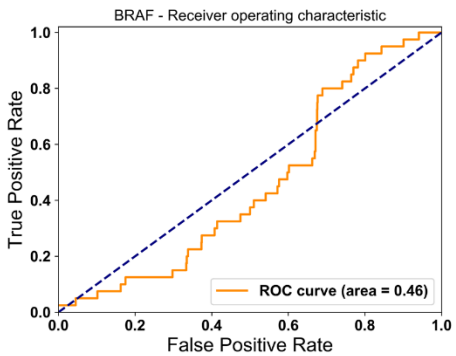


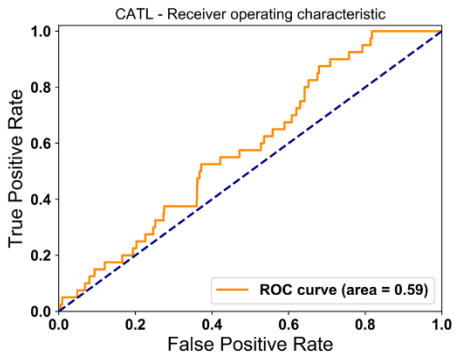


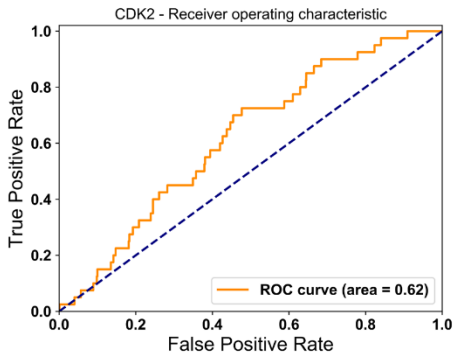


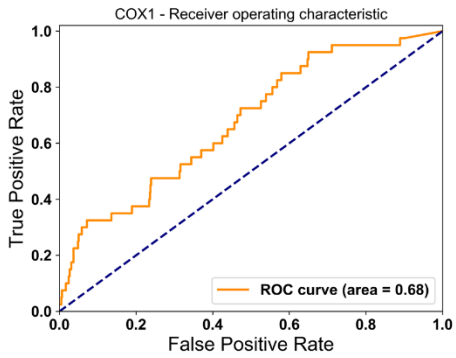


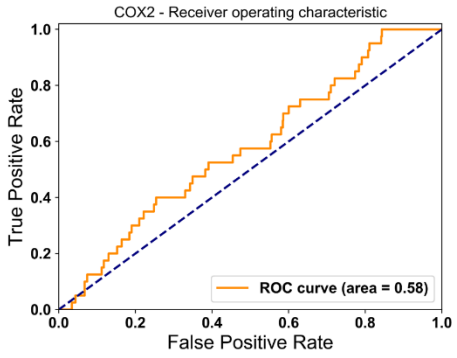


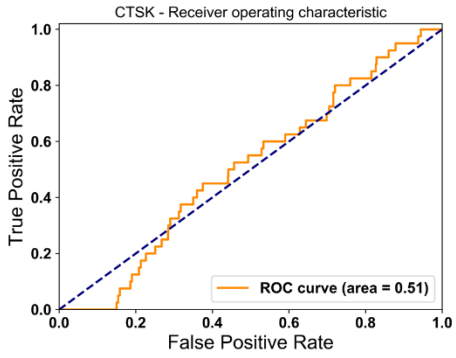


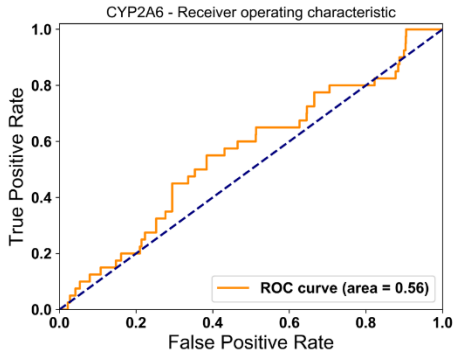

CATL - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

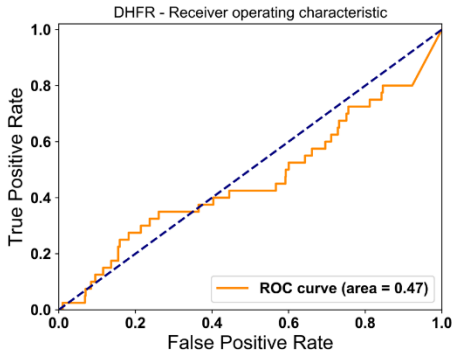

CDK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

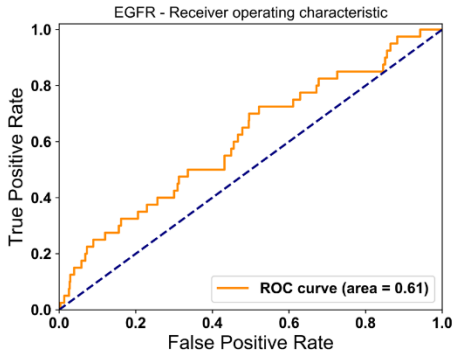

COX1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.68)

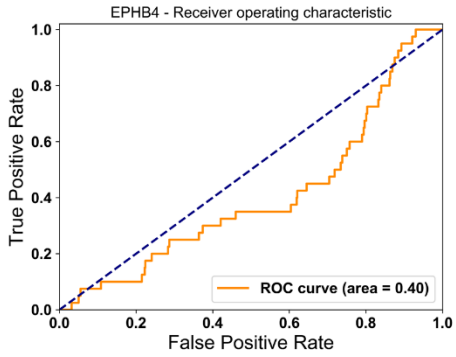

COX2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

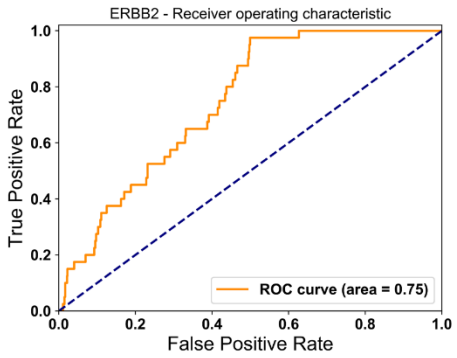

CTSK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

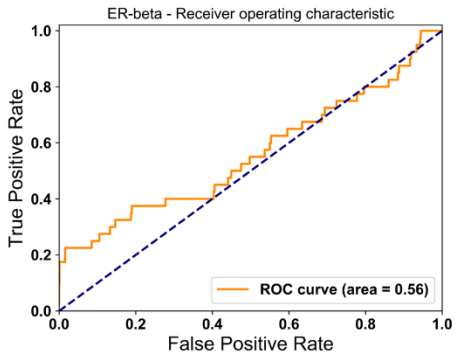

CYP2A6 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

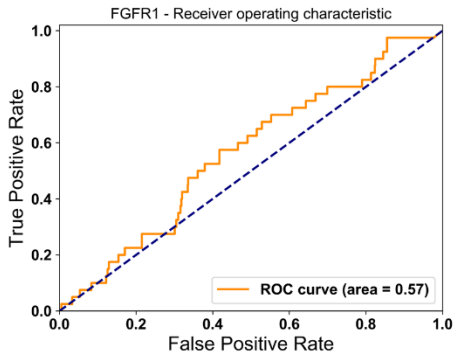

DHFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

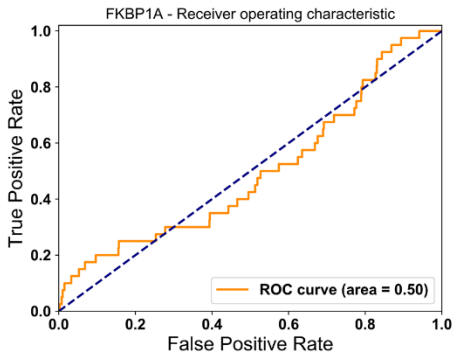

EGFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

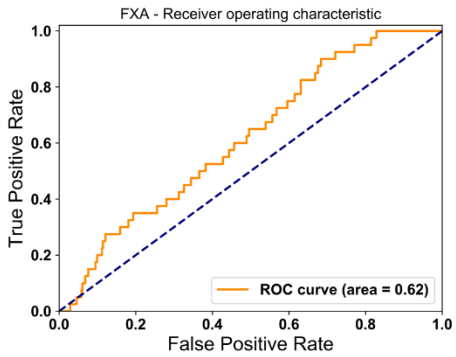

EPHB4 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.40)

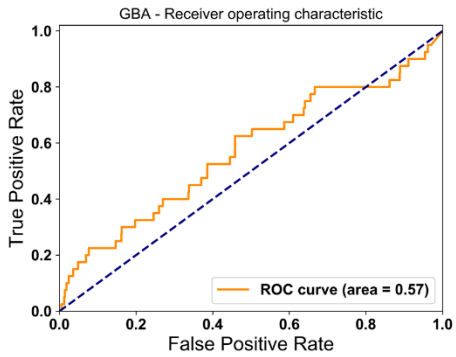

ERBB2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)

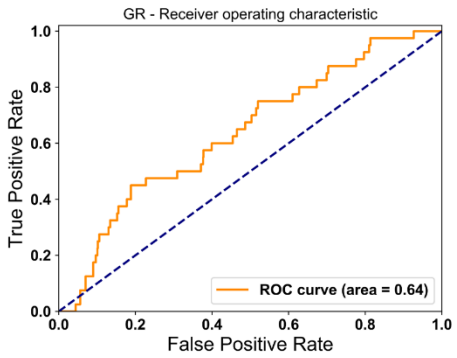

ER-beta - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

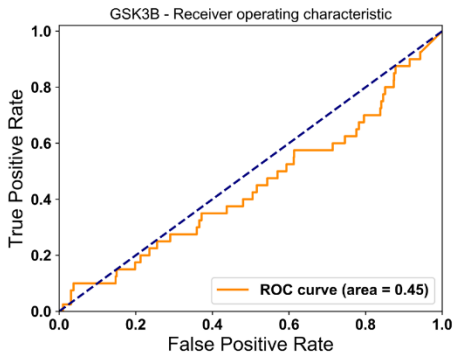

FGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

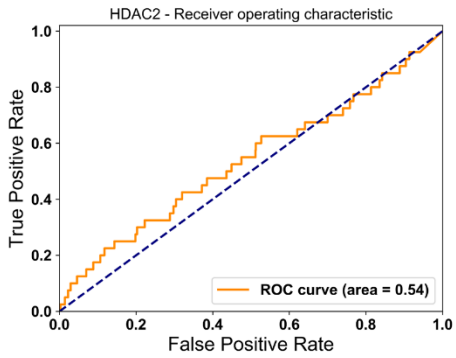

FKBP1A - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.50)

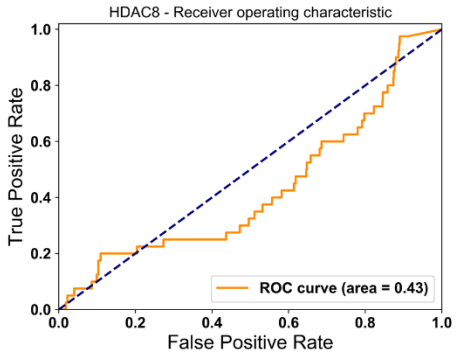

FXA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

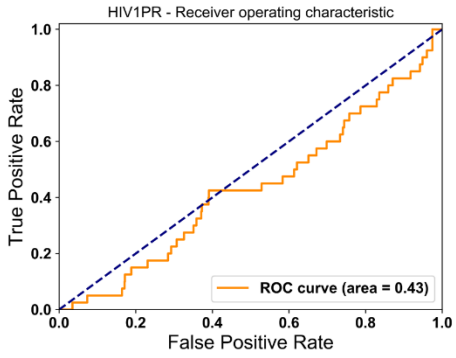

GBA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

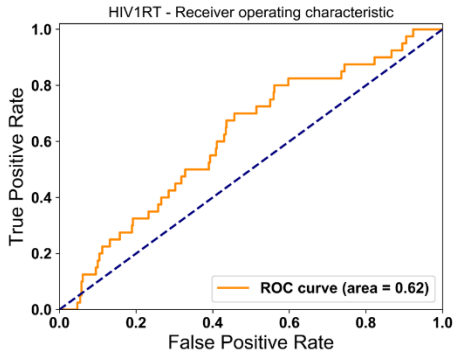

GR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

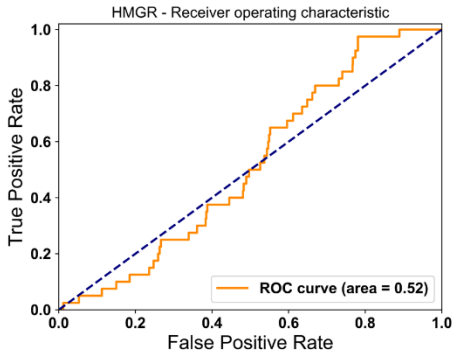

GSK3B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

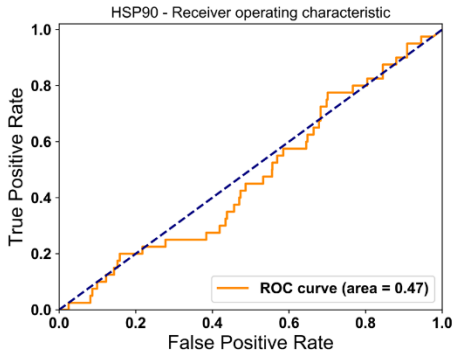

HDAC2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

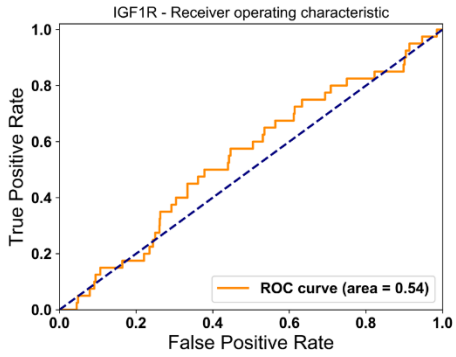

HDAC8 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.43)

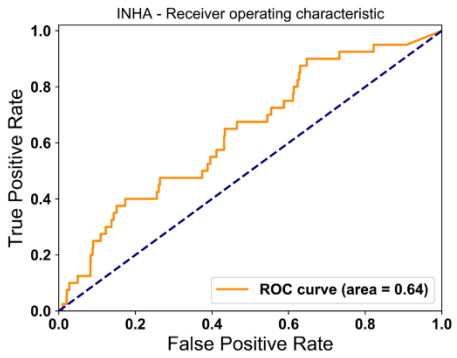

HIV1PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.43)

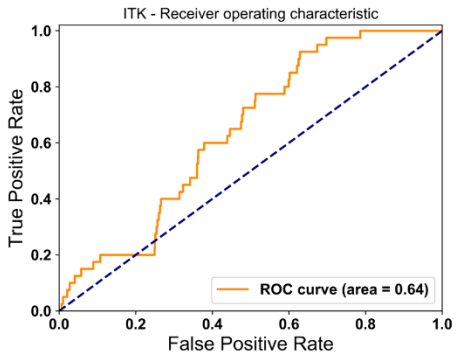

HIV1RT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

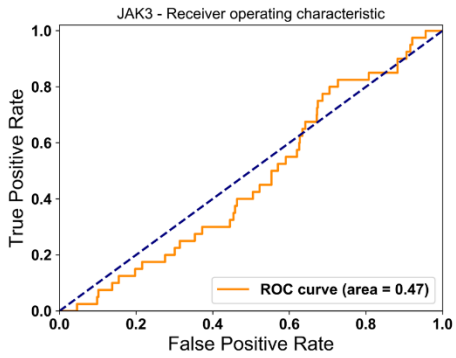

HMGR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

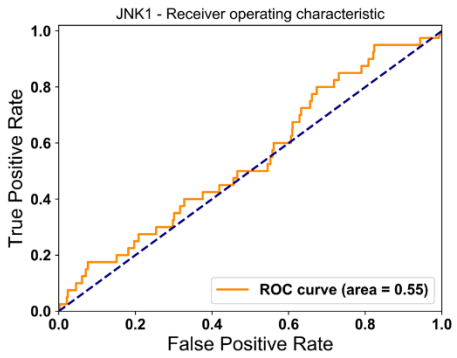

HSP90 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

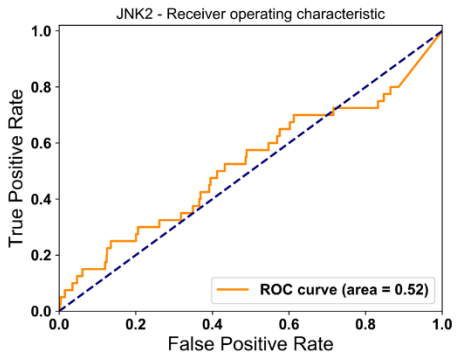

IGF1R - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

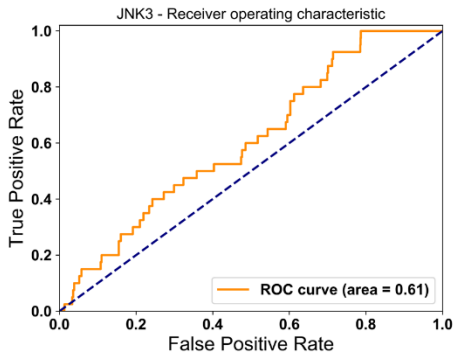

INHA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

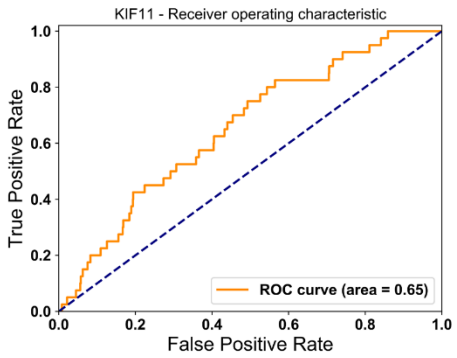

ITK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

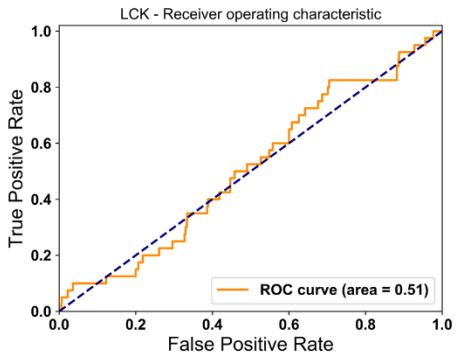

JAK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

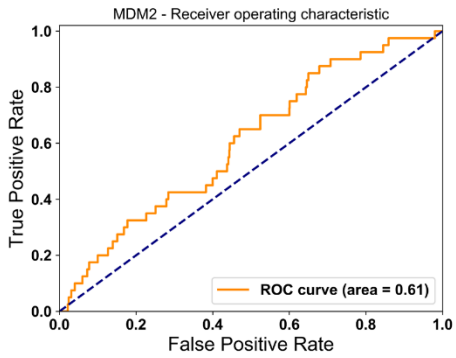

JNK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

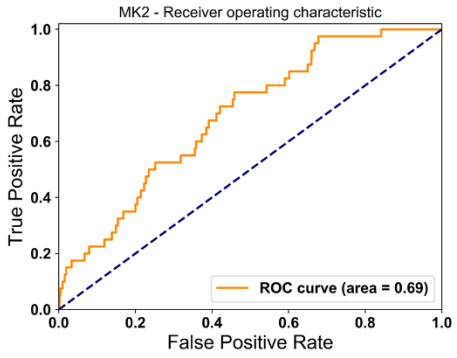

JNK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

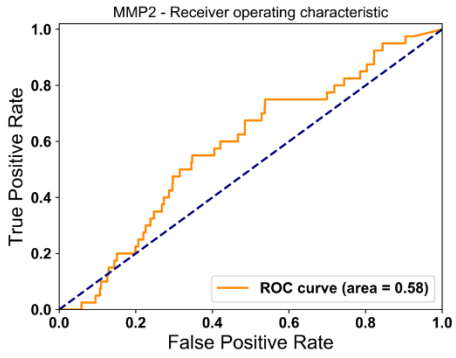

JNK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

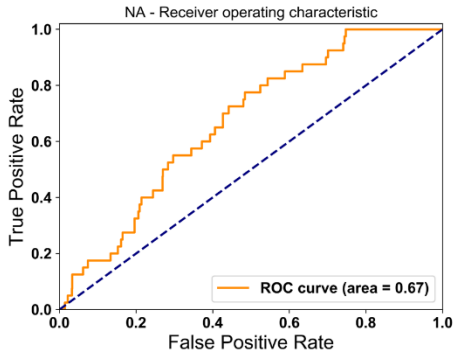

KIF11 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

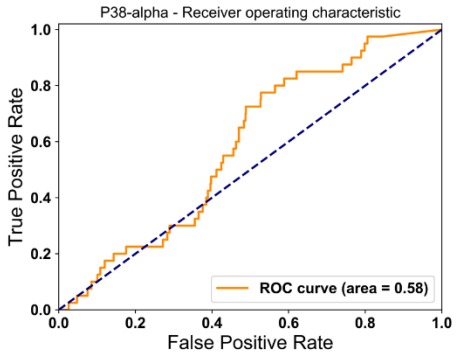

LCK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

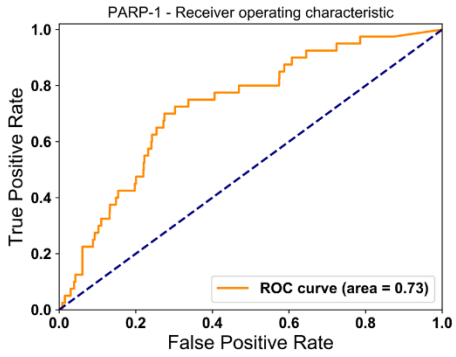

MDM2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

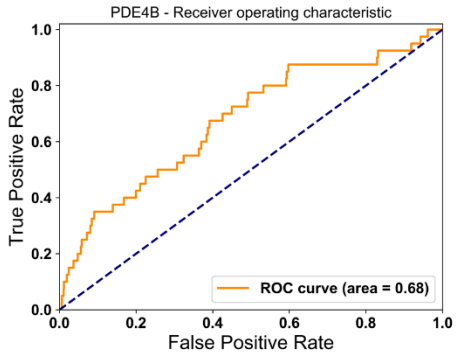

MK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

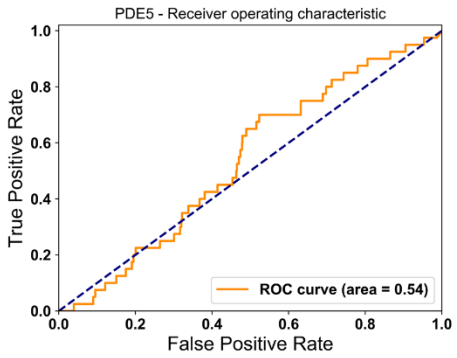

MMP2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

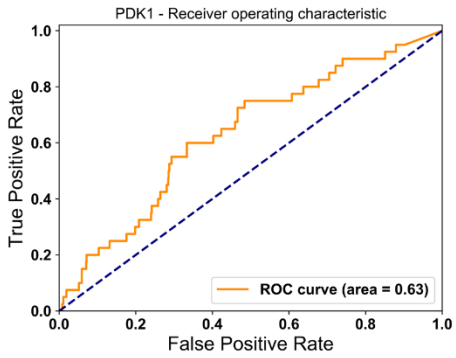

NA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

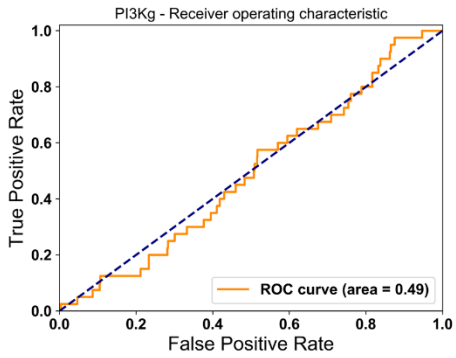

P38-alpha - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

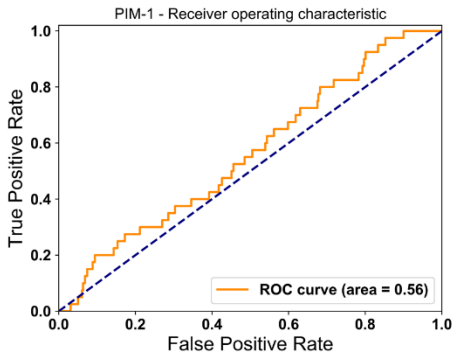

PARP-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.73)

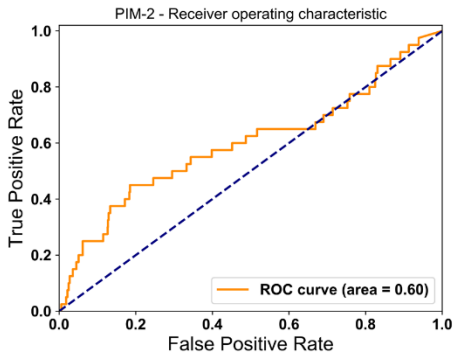

PDE4B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.68)

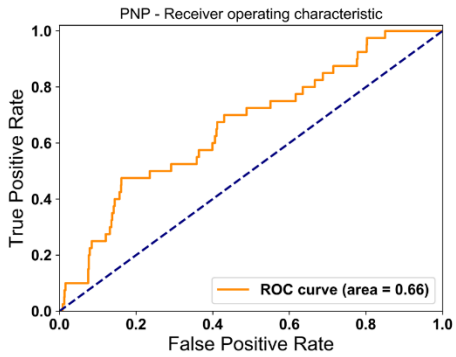

PDE5 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

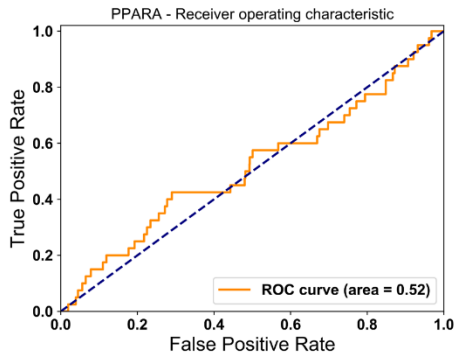

PDK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

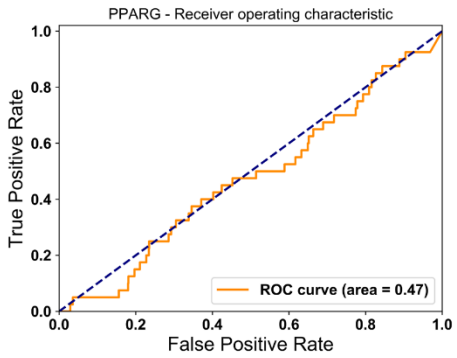

PI3Kg - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.49)

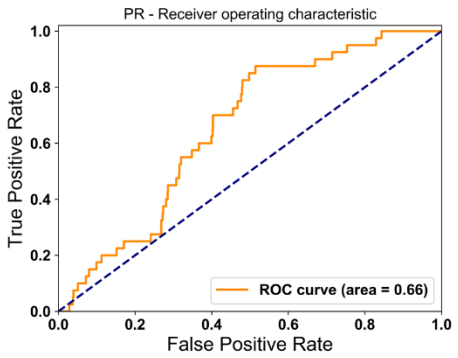

PIM-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

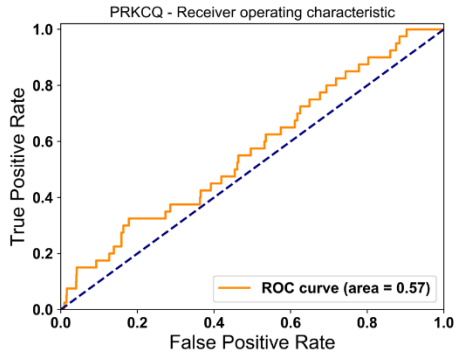

PIM-2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

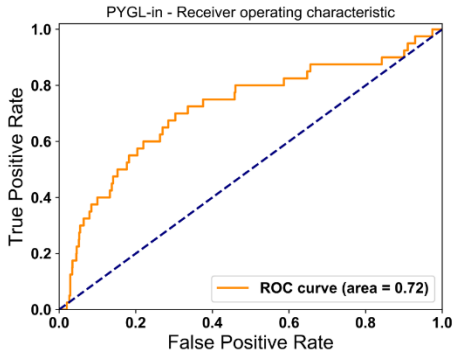

PNP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

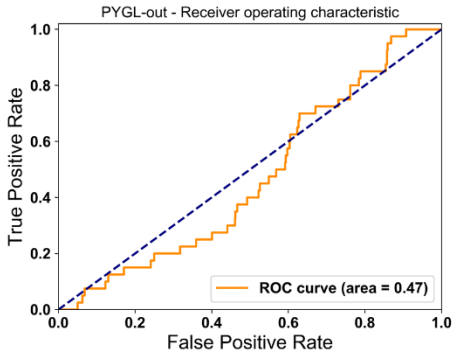

PPARA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

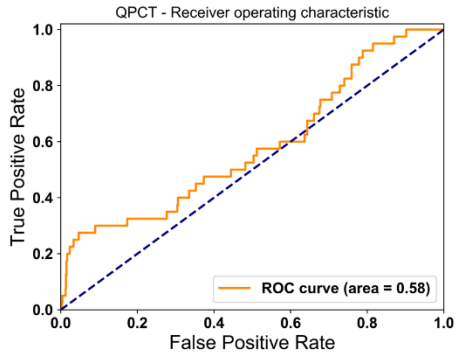

PPARG - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

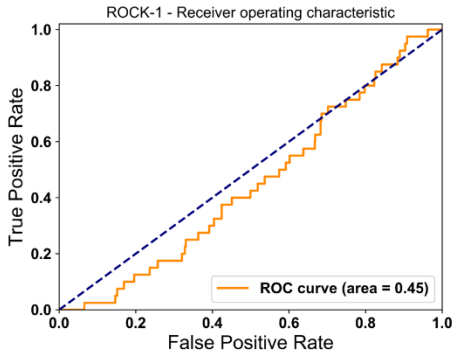

PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

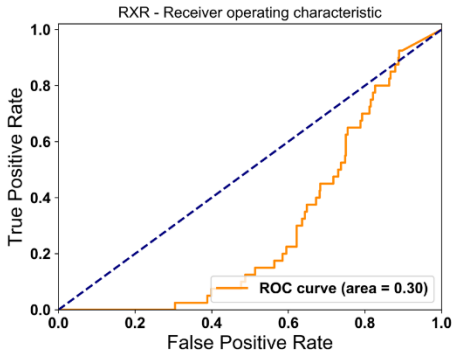

PRKCQ - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

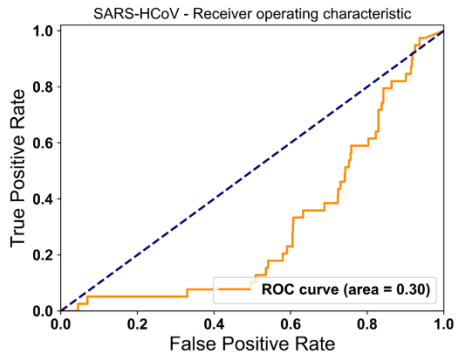

PYGL-in - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

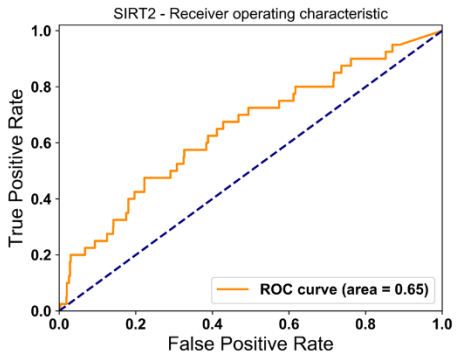

PYGL-out - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

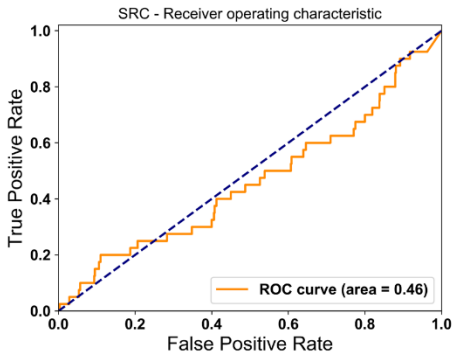

QPCT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

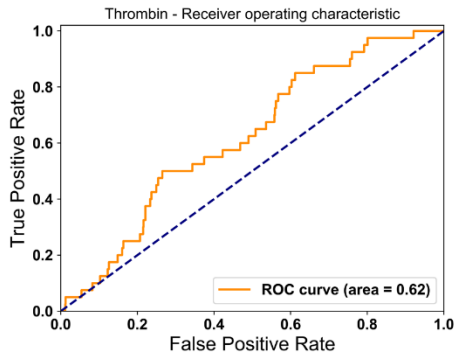

ROCK-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

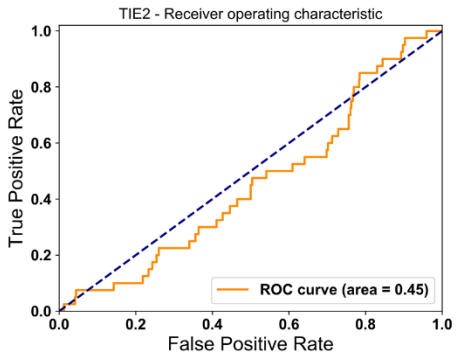

RXR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.30)

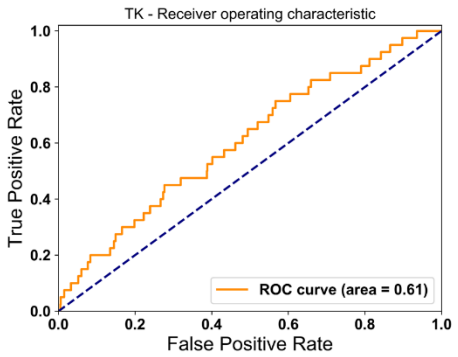

SARS-HCoV - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.30)

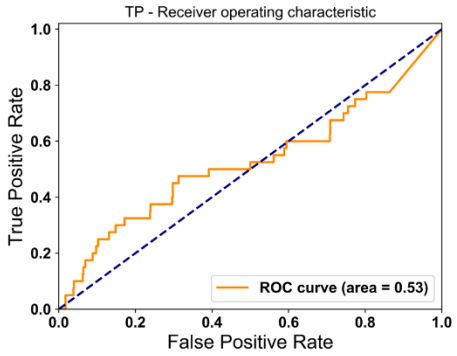

SIRT2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

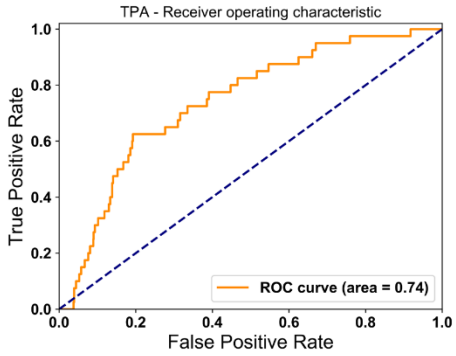

SRC - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

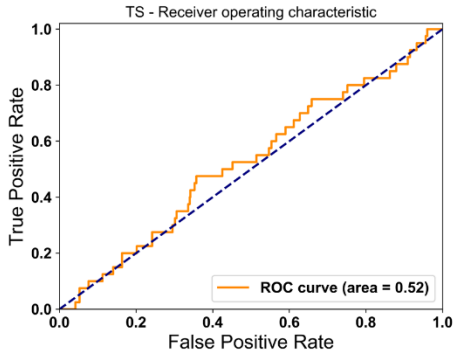

Thrombin - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

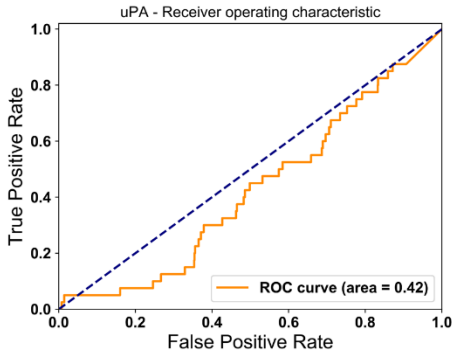

TIE2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

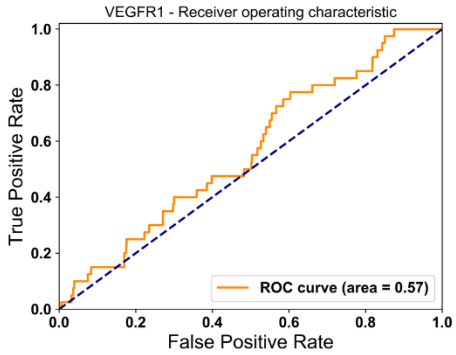

TK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

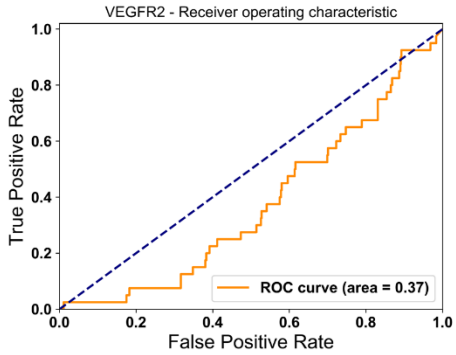

TP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)


TPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)


TS - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

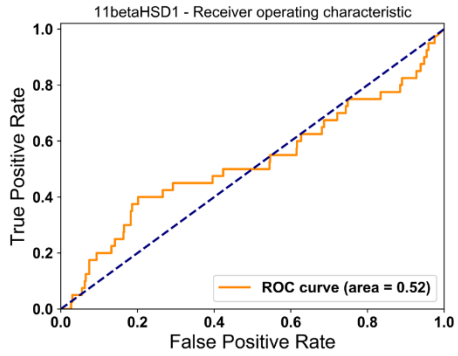

uPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.42)

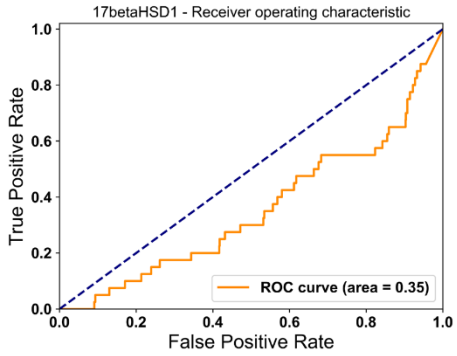

VEGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

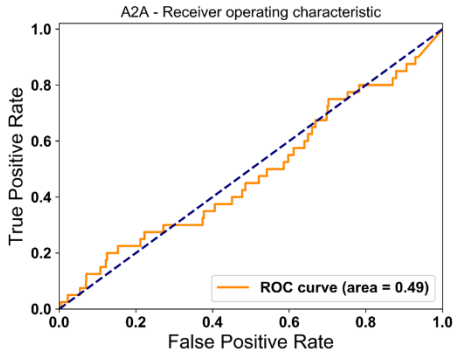

VEGFR2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.37)

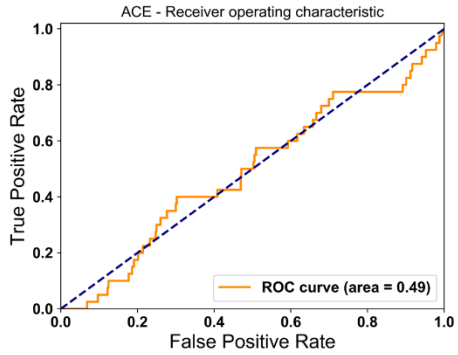
ROC of LIDAEUS flexible-docking results

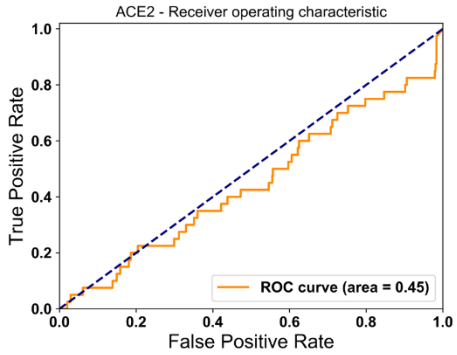
(Conformers generated by OpenBabel)

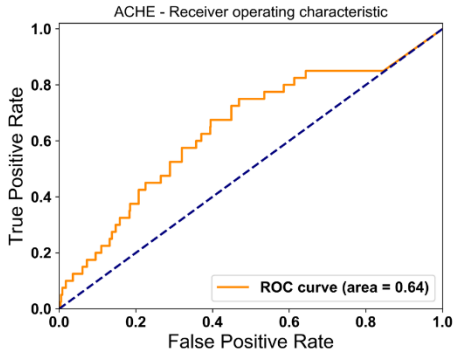


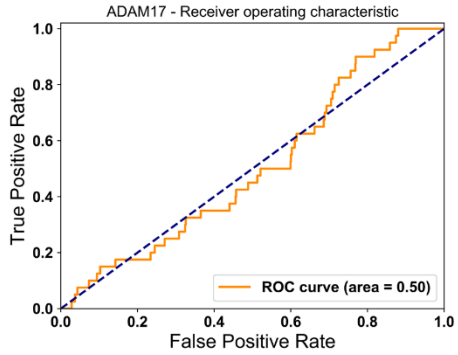


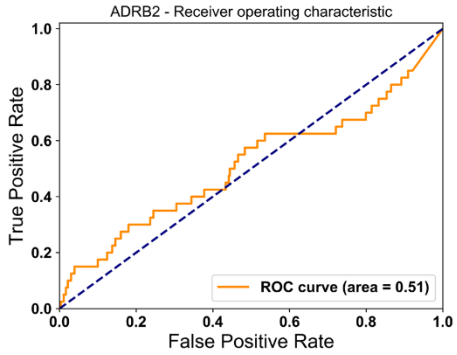


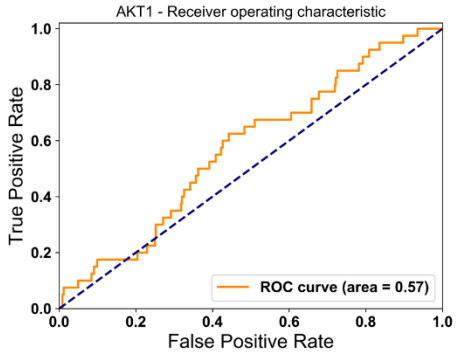


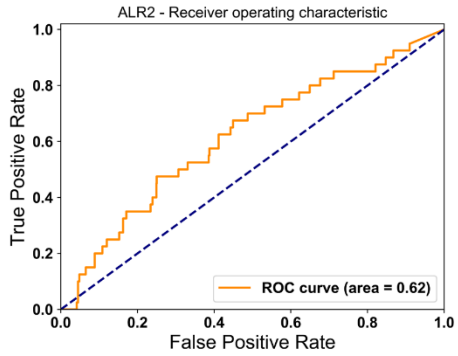


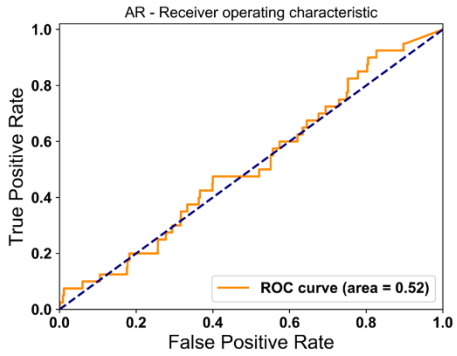


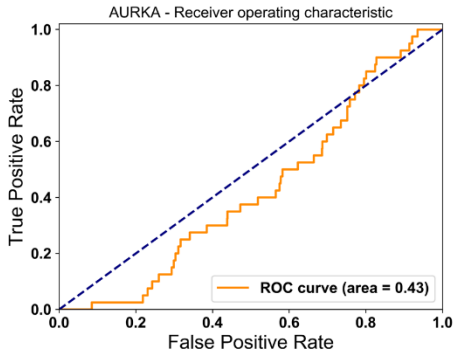


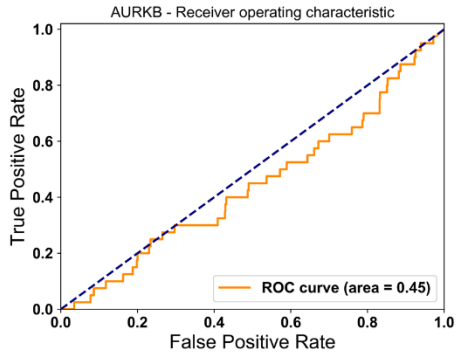


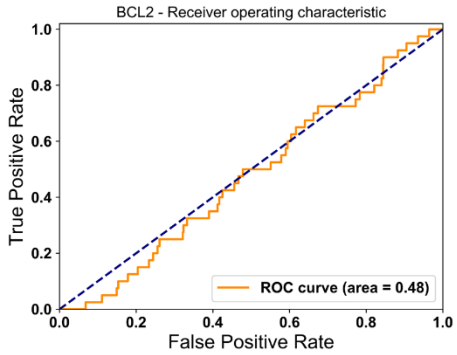


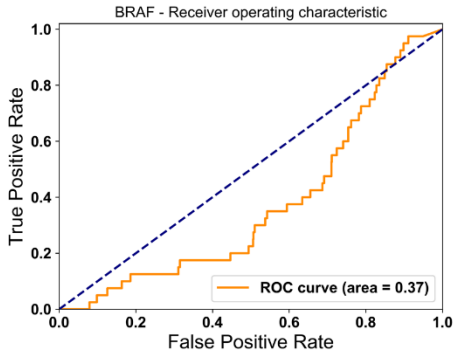


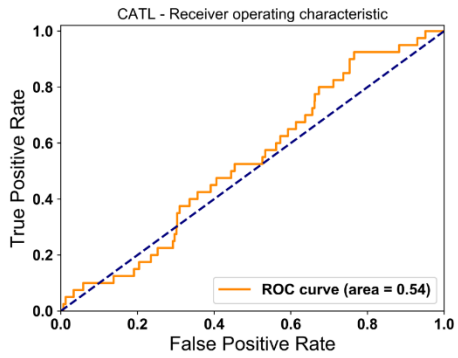


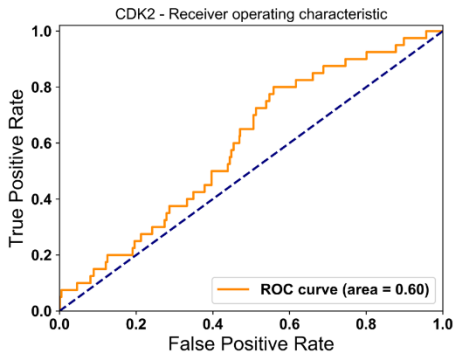


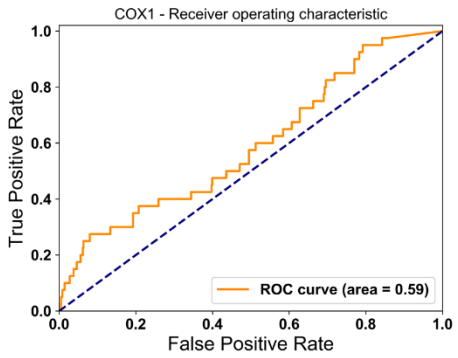


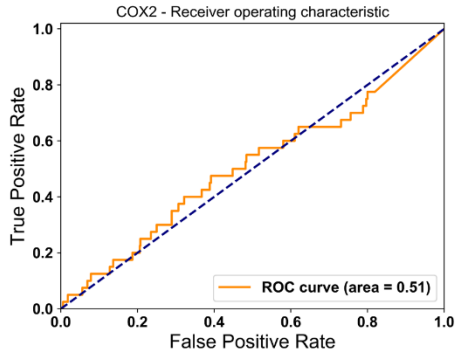


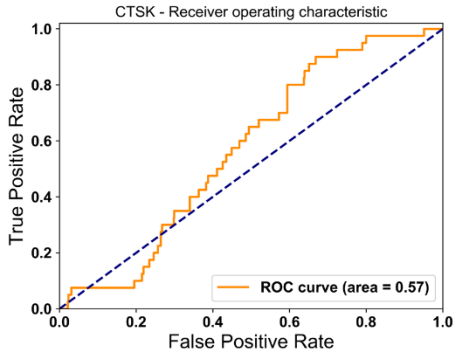


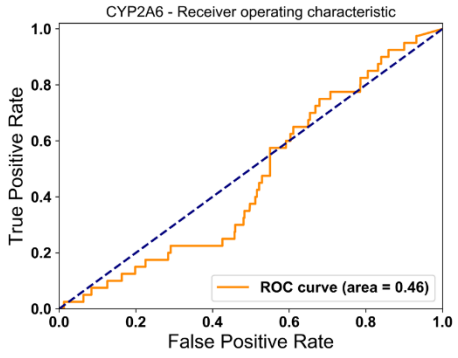

CATL - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

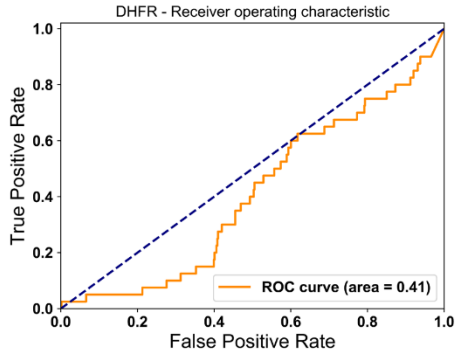

CDK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

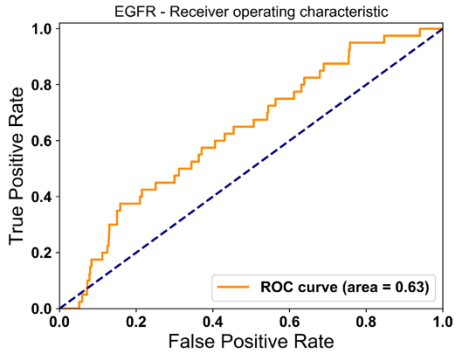

COX1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

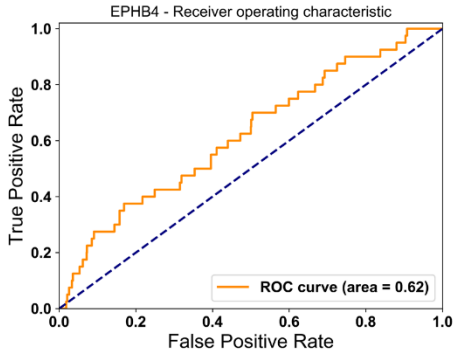

COX2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

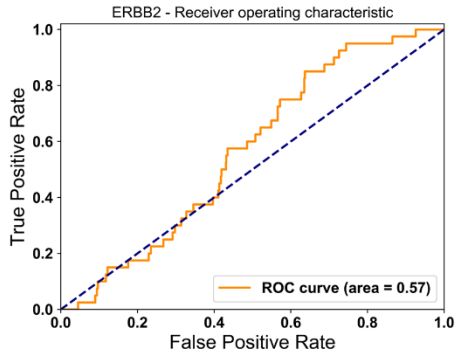

CTSK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

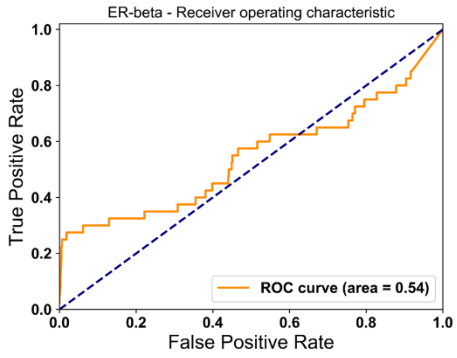

CYP2A6 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

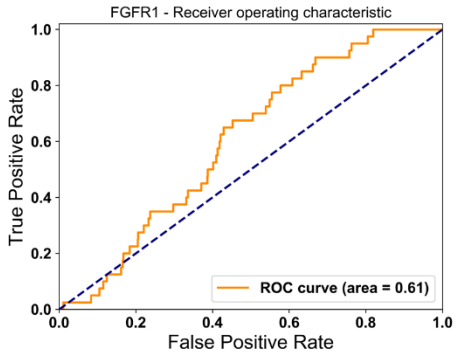

DHFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.41)

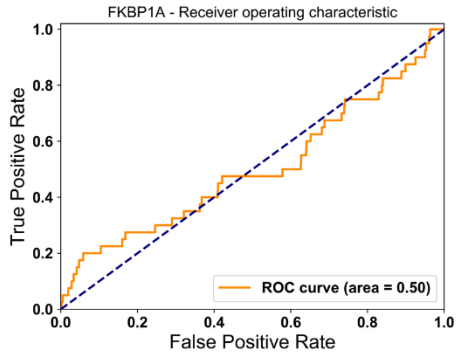

EGFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

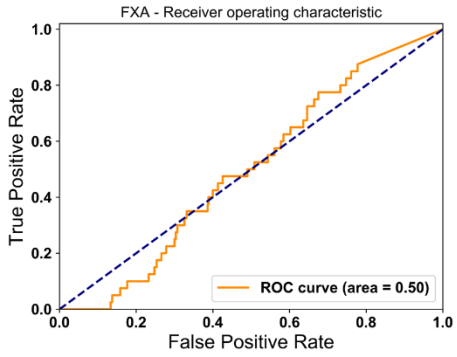

EPHB4 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

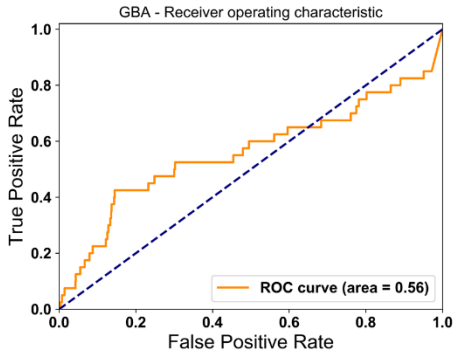

ERBB2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

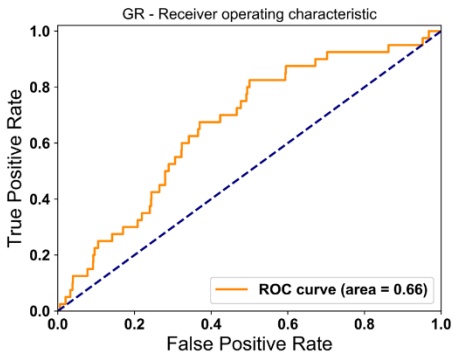

ER-beta - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

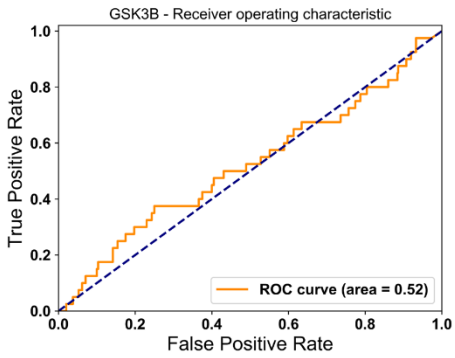

FGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

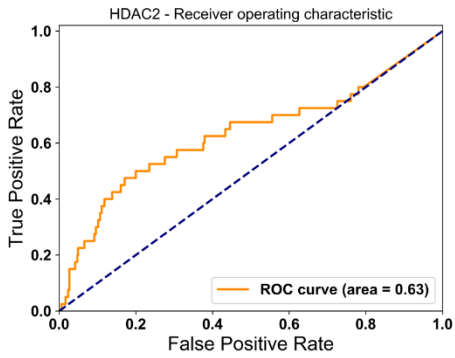

FKBP1A - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.50)

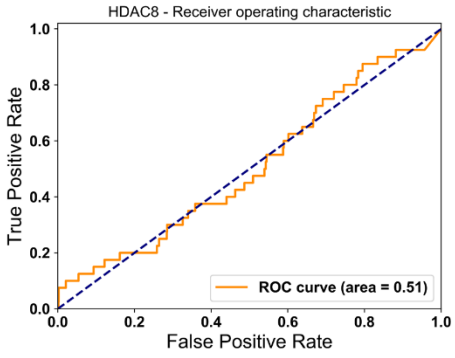

FXA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.50)

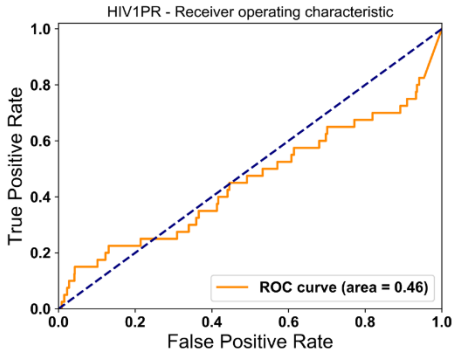

GBA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

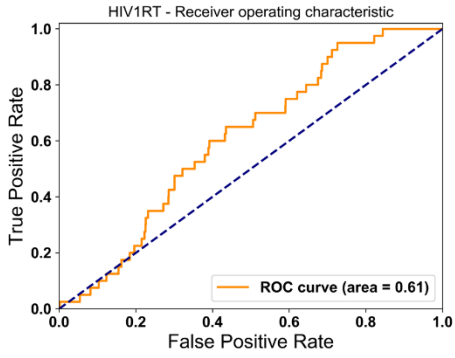

GR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

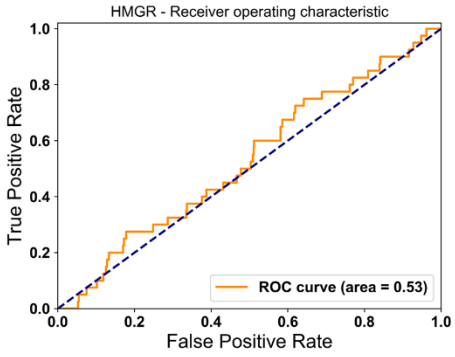

GSK3B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

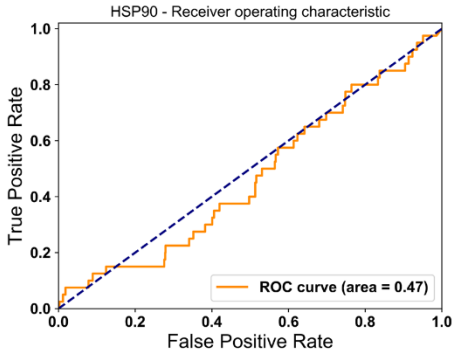

HDAC2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

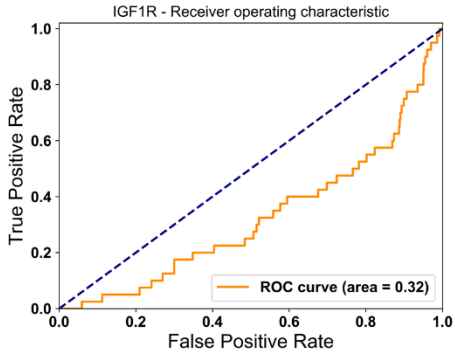

HDAC8 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

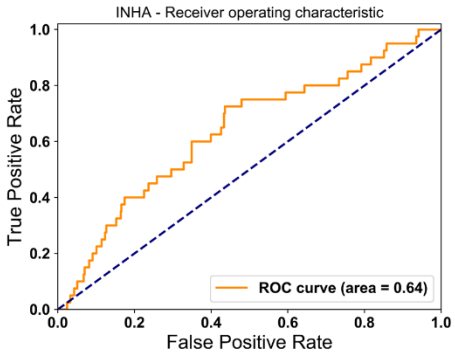

HIV1PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

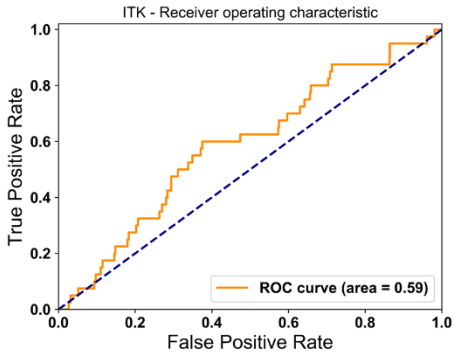

HIV1RT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

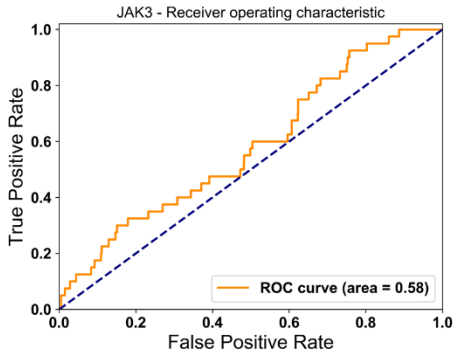

HMGR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

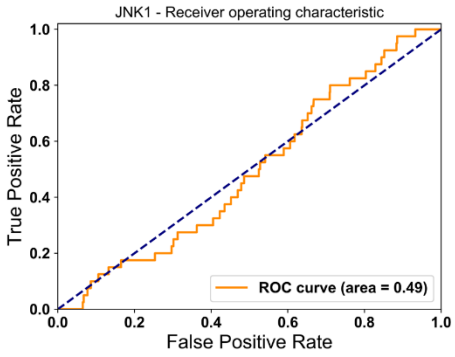

HSP90 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

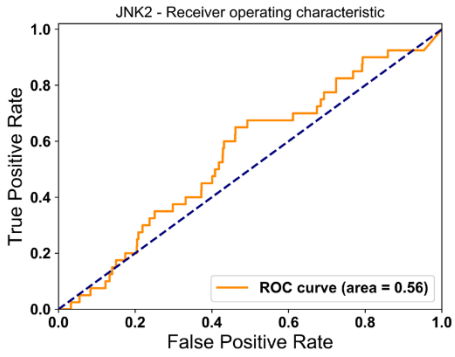

IGF1R - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.32)

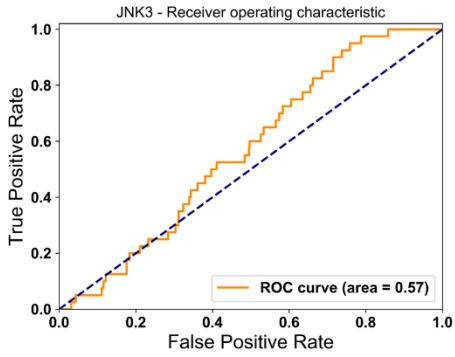

INHA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

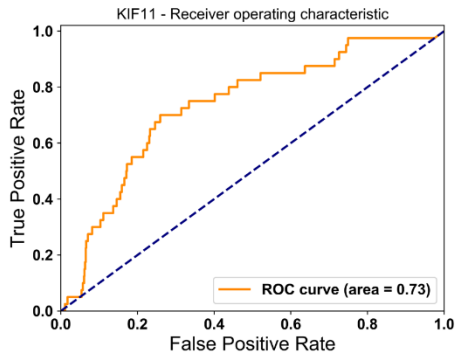

ITK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

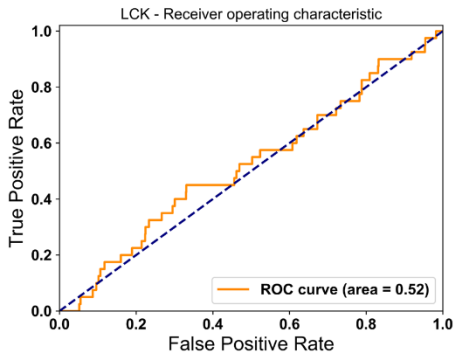

JAK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

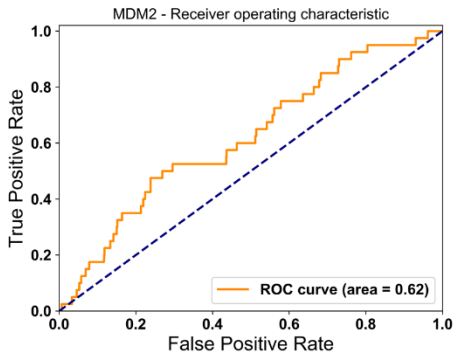

JNK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.49)

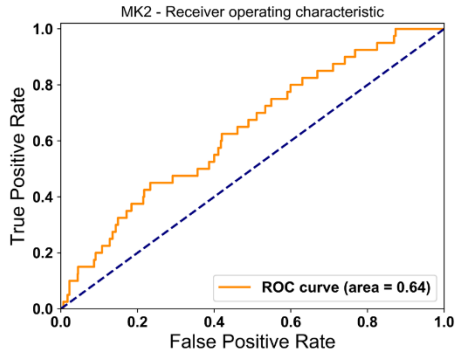

JNK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

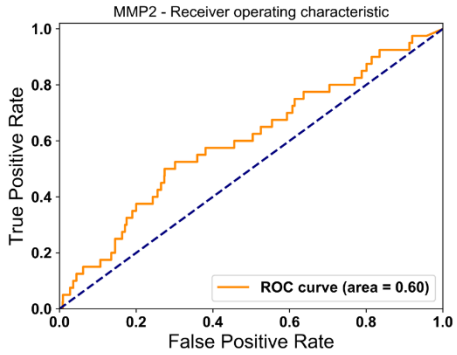

JNK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

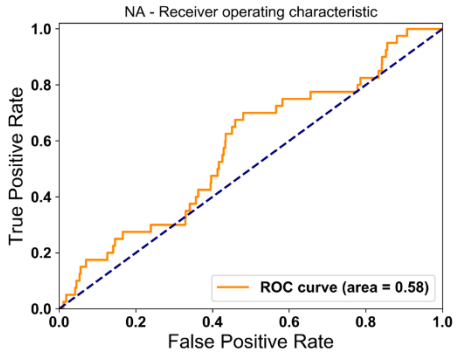

KIF11 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.73)

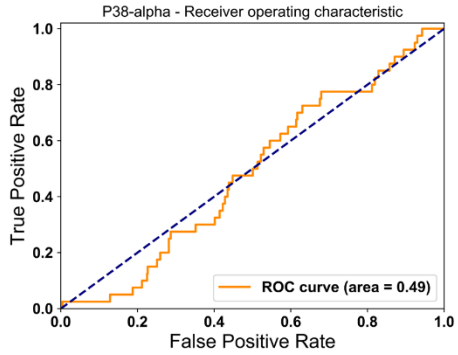

LCK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

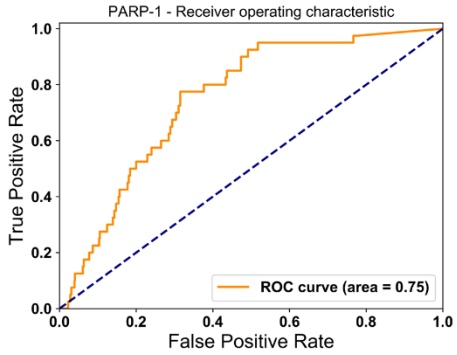

MDM2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

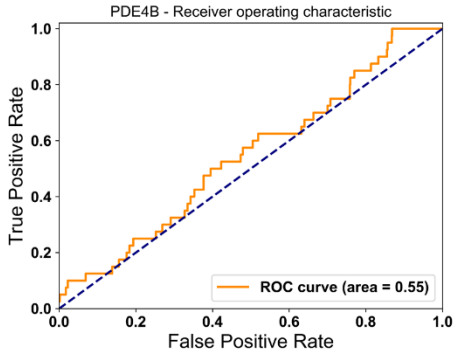

MK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

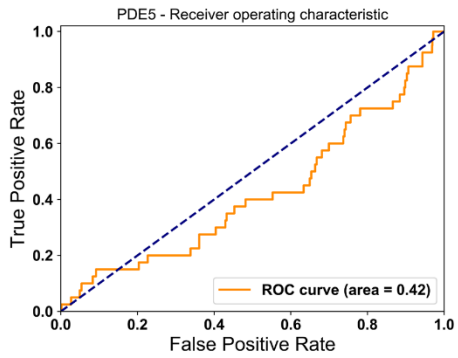

MMP2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

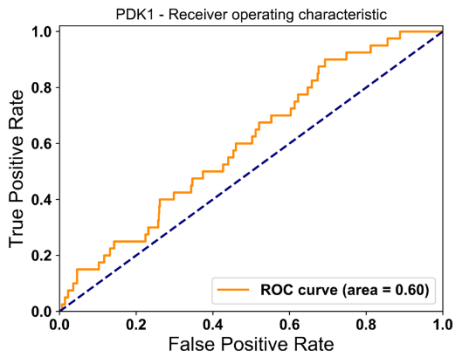

NA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

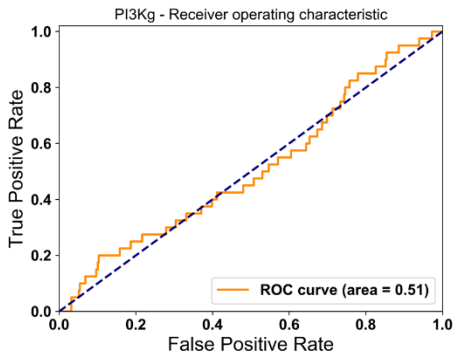

P38-alpha - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.49)

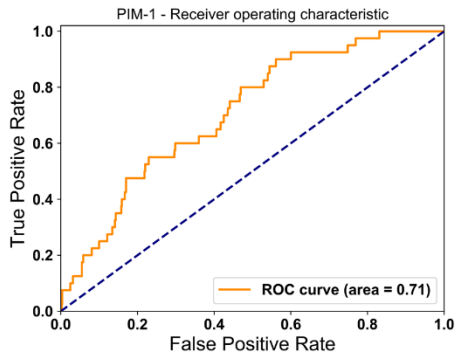

PARP-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)

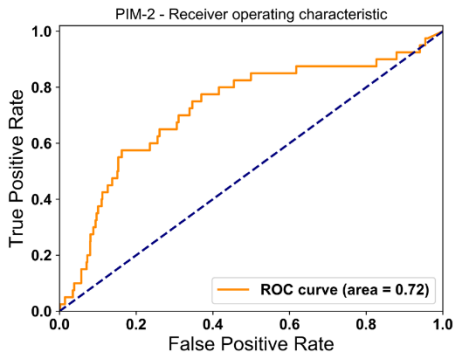

PDE4B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

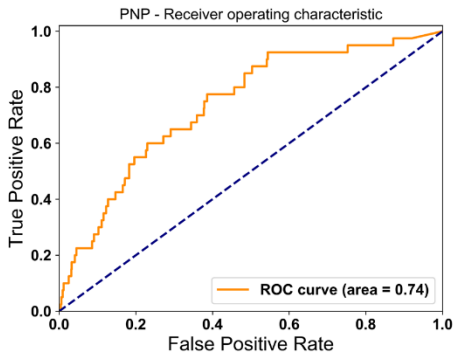

PDE5 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.42)

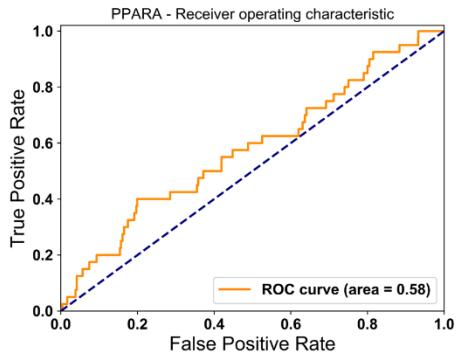

PDK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

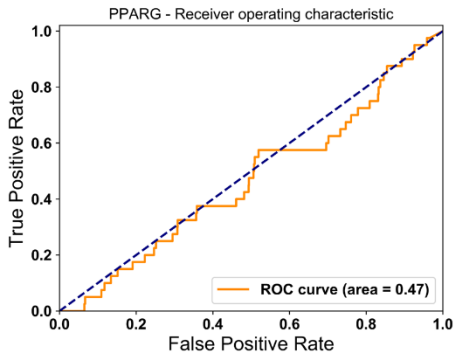

PI3Kg - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

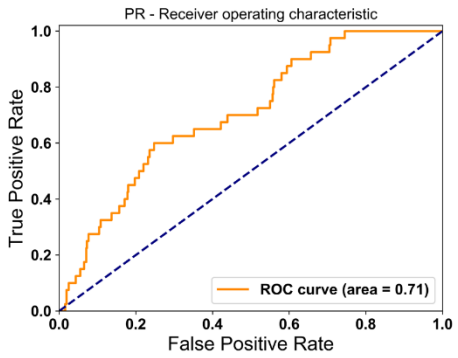

PIM-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

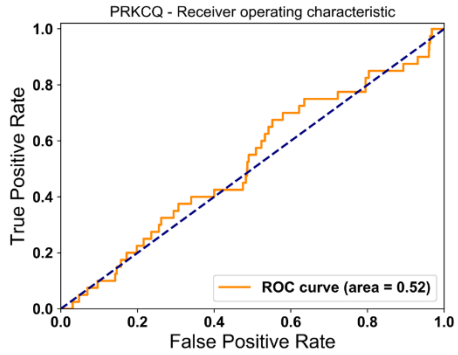

PIM-2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

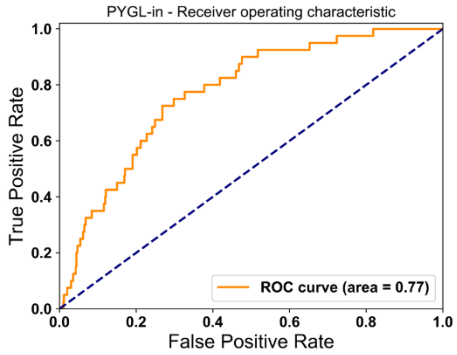

PNP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

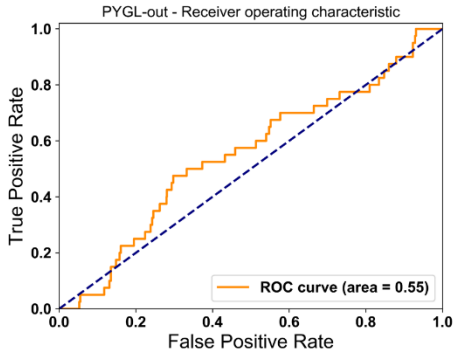

PPARA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

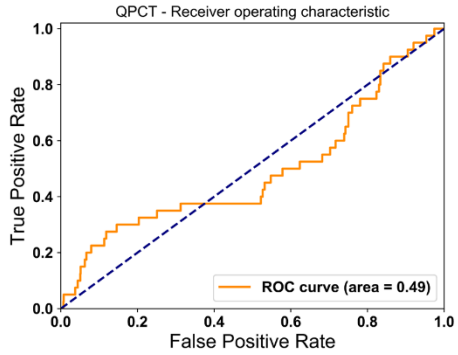

PPARG - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

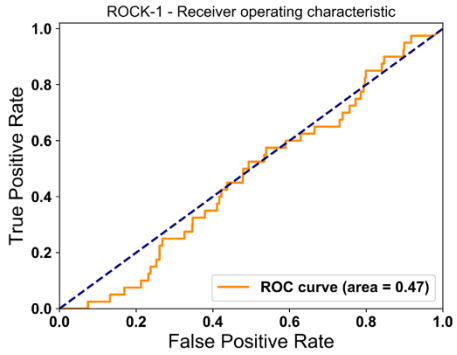

PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

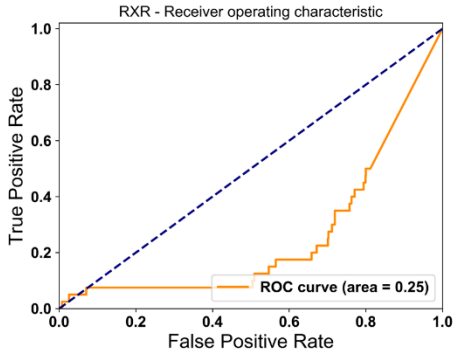

PRKCQ - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

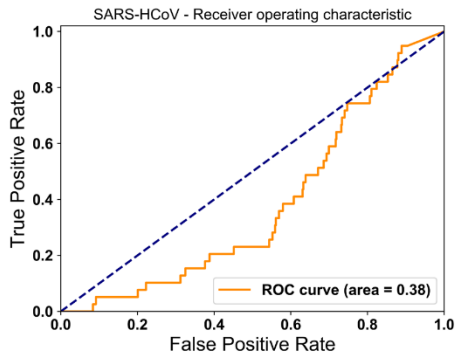

PYGL-in - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)

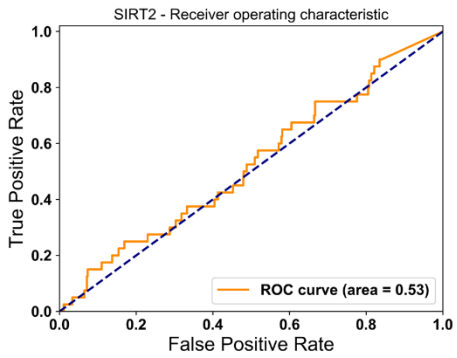

PYGL-out - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

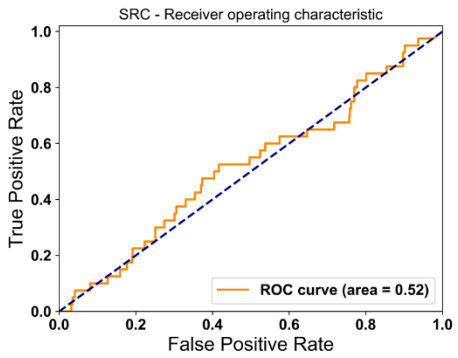

QPCT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.49)

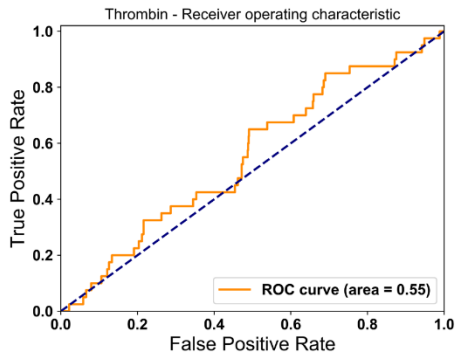

ROCK-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

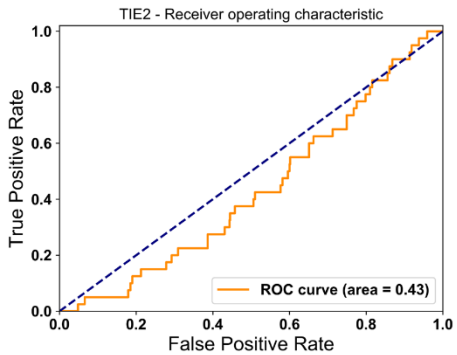

RXR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.25)

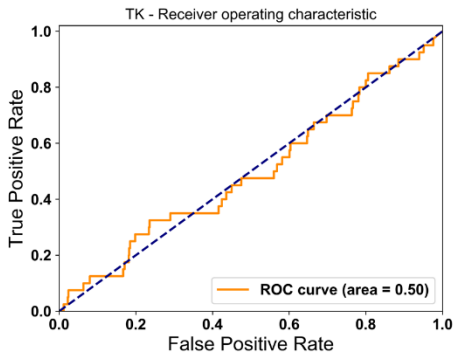

SARS-HCoV - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.38)

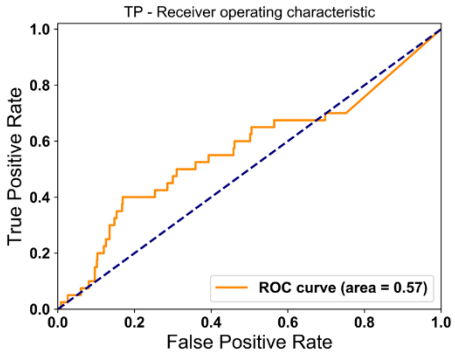

SIRT2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

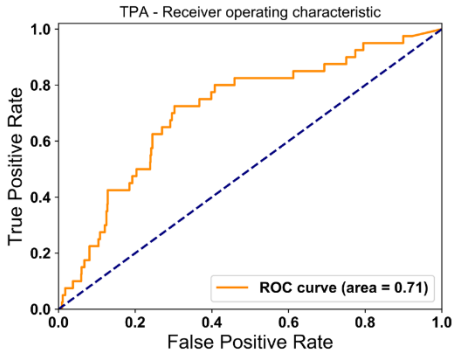

SRC - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

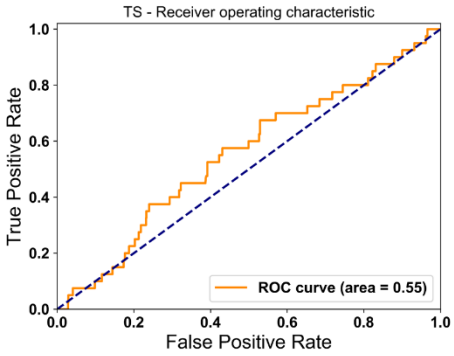

Thrombin - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

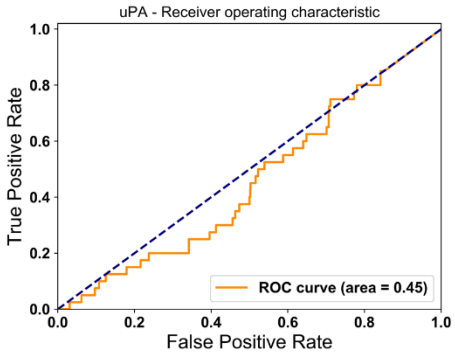

TIE2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.43)

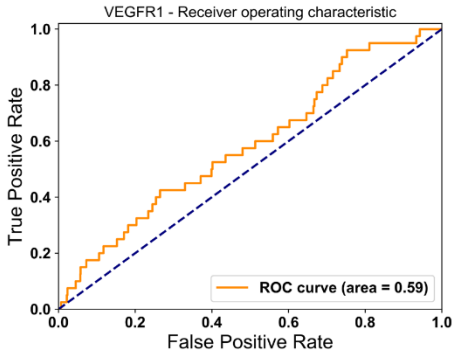

TK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.50)

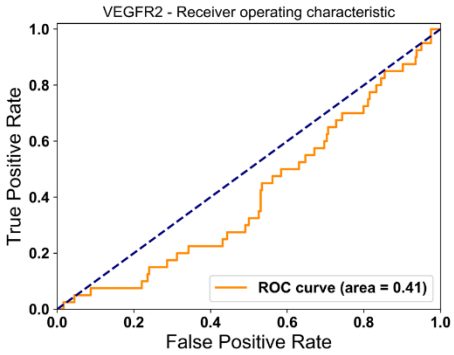

TP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)


TPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

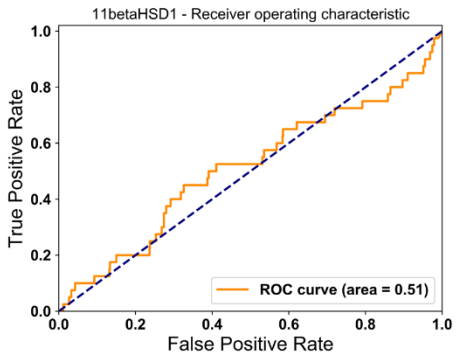

TS - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

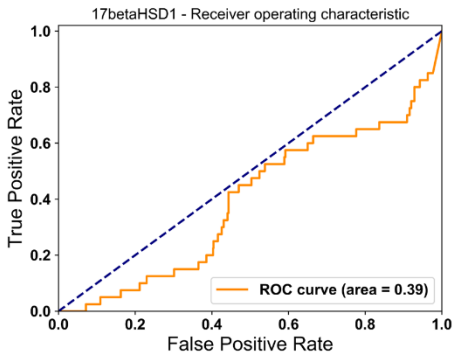

uPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

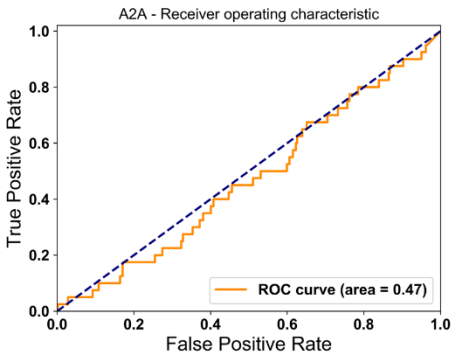

VEGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

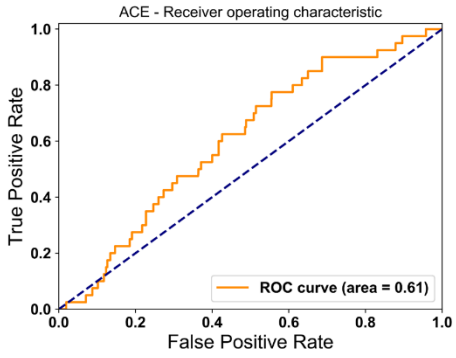

VEGFR2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.41)

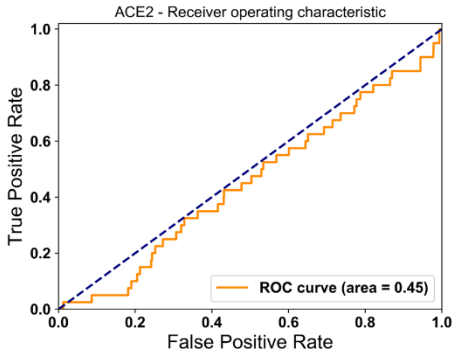
ROC of LIDAEUS flexible-docking result

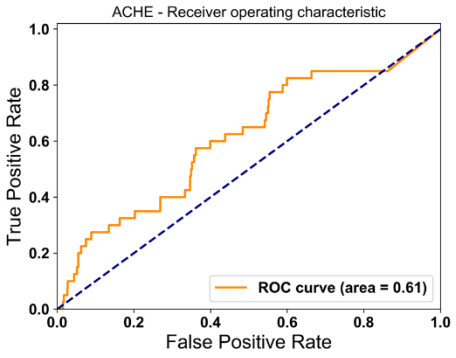


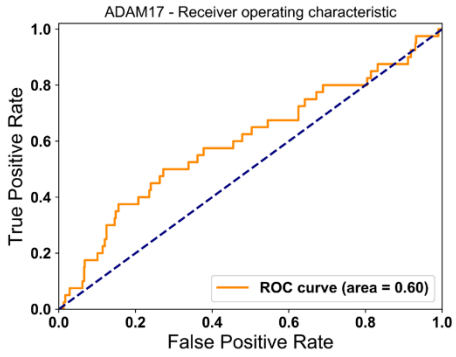


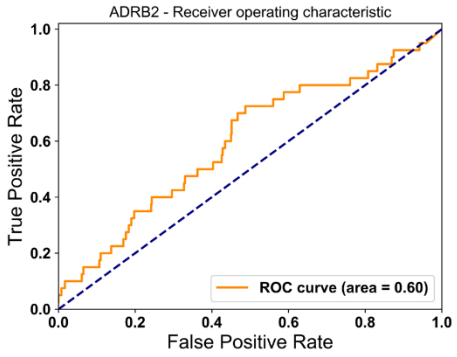


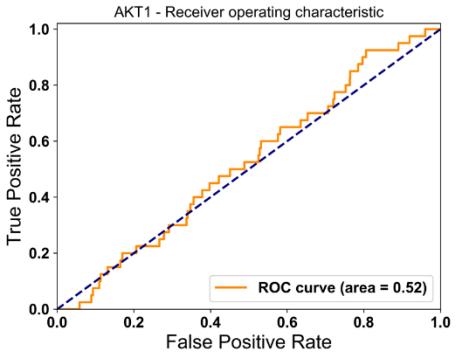


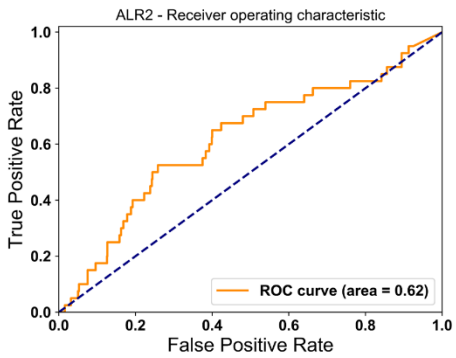


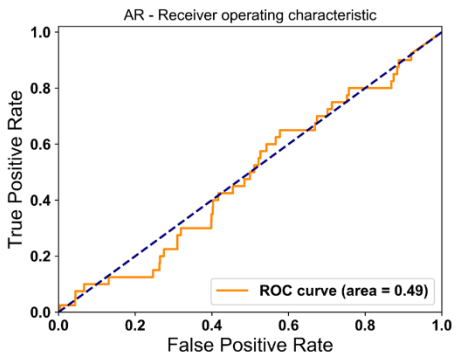


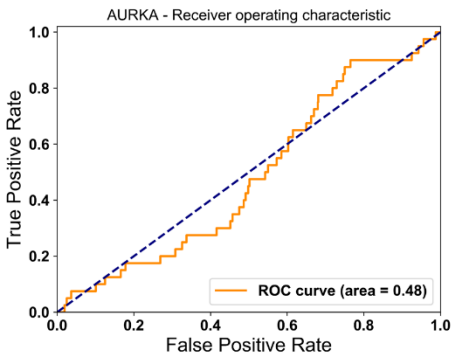


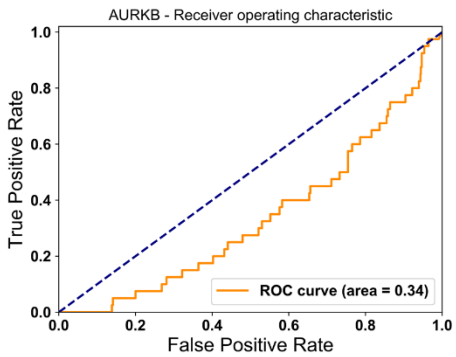


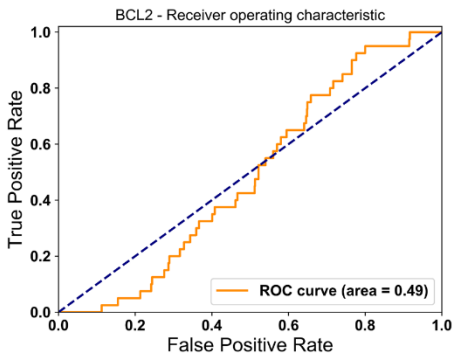


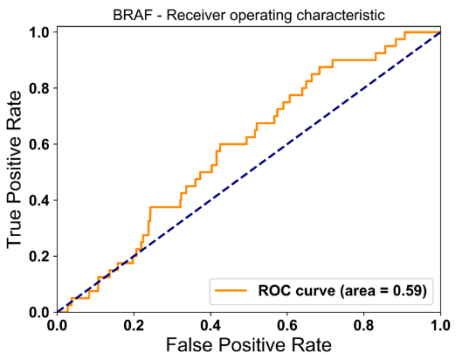


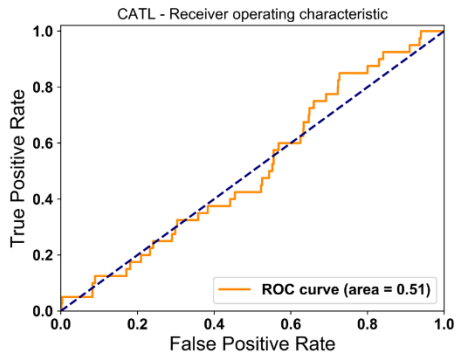


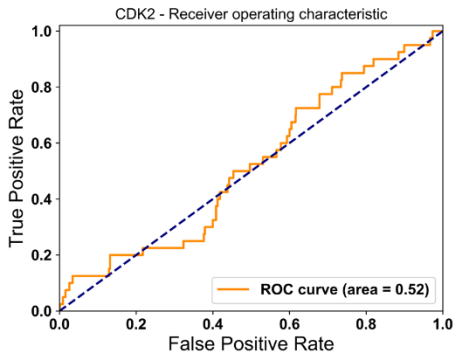


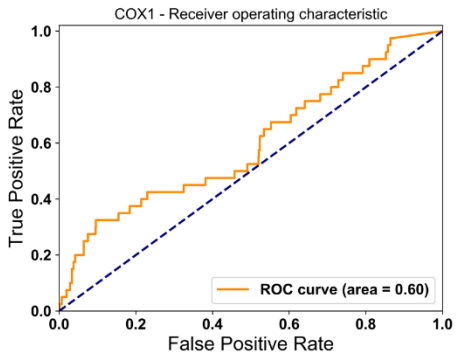


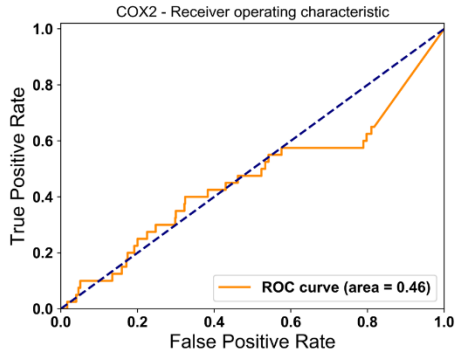


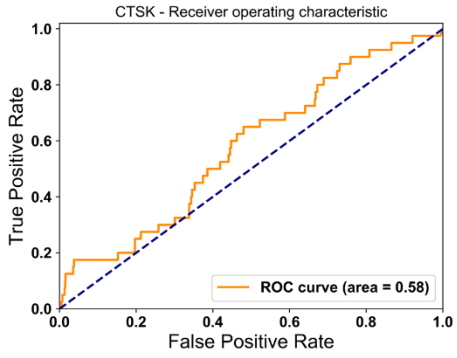


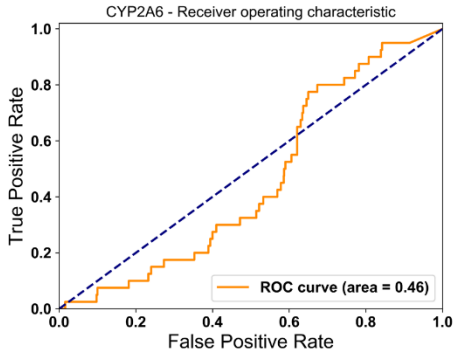

CATL - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

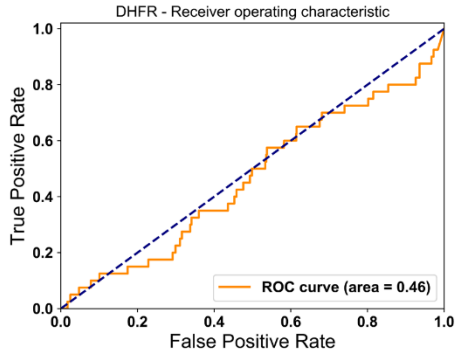

CDK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

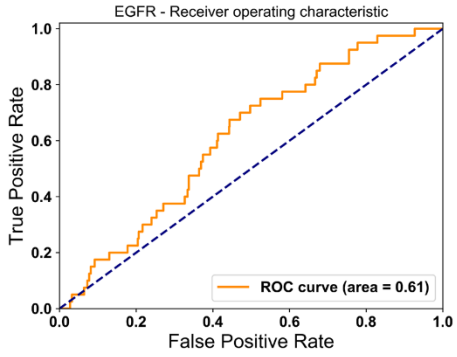

COX1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

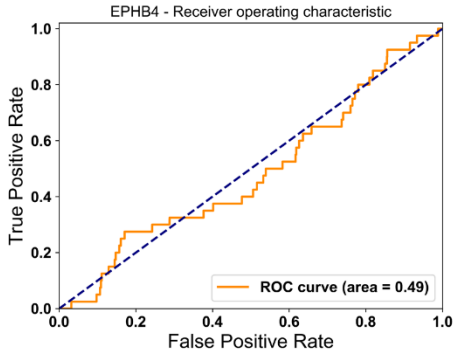

COX2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

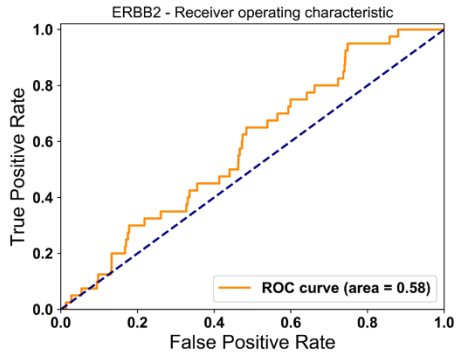

CTSK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

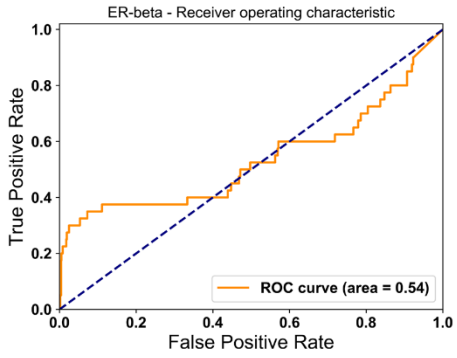

CYP2A6 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

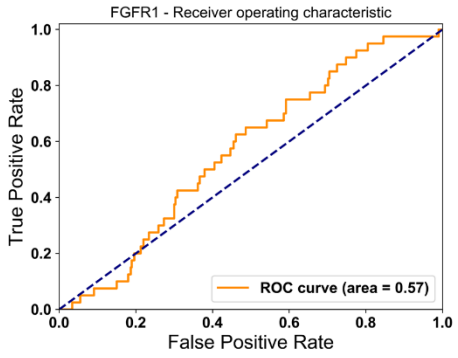

DHFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

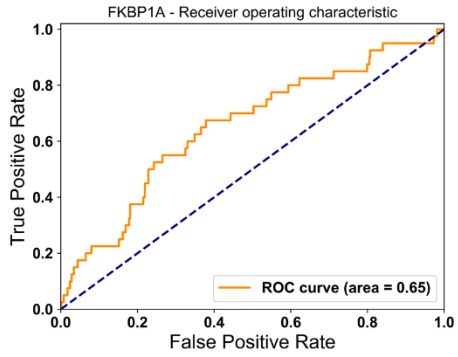

EGFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

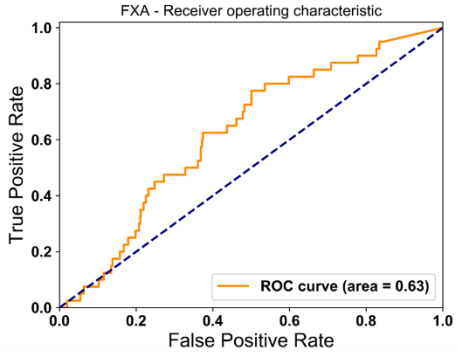

EPHB4 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.49)

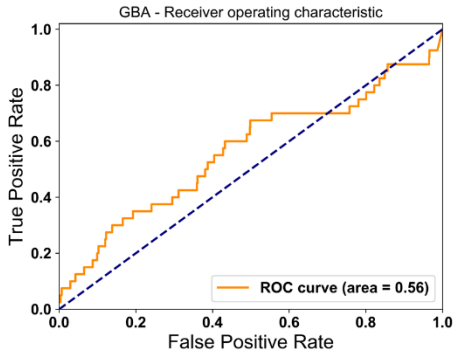

ERBB2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.58)

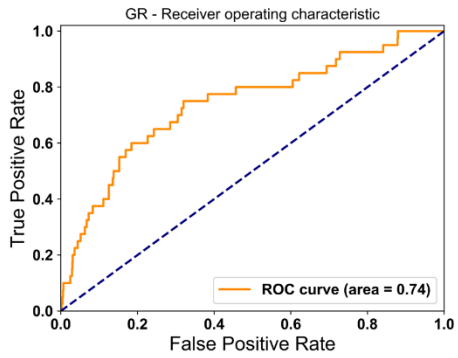

ER-beta - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

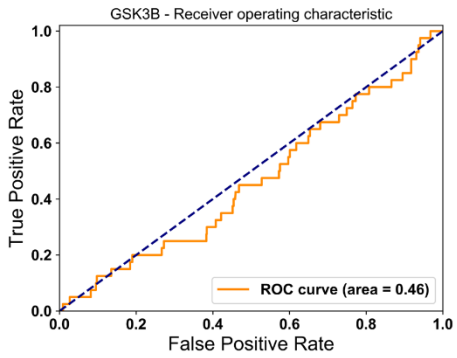

FGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

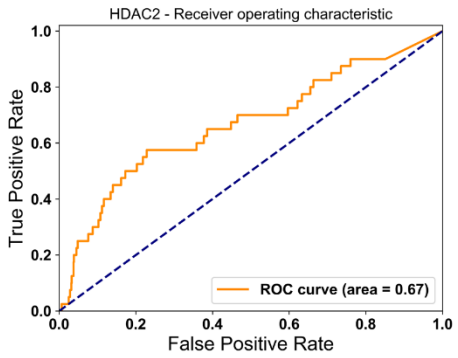

FKBP1A - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

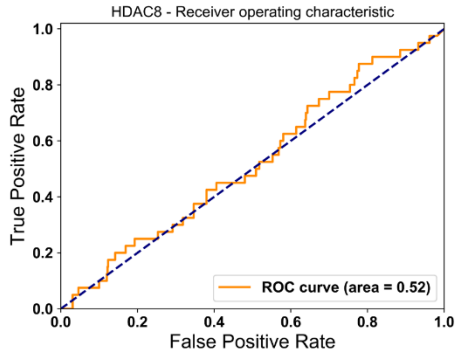

FXA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

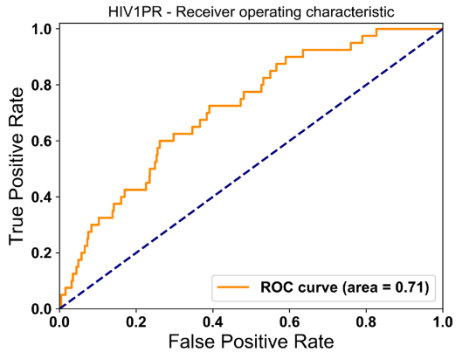

GBA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

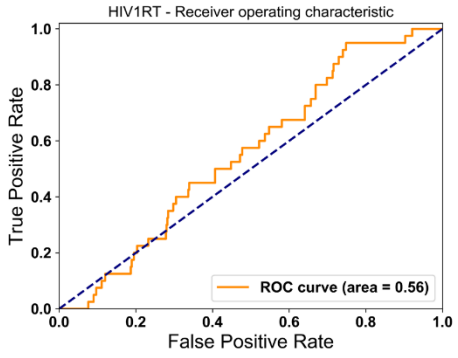

GR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

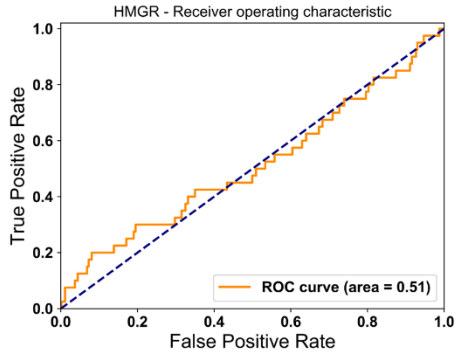

GSK3B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

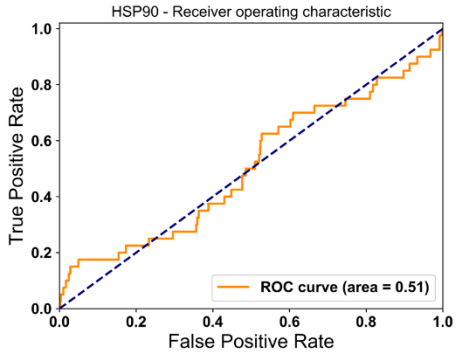

HDAC2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

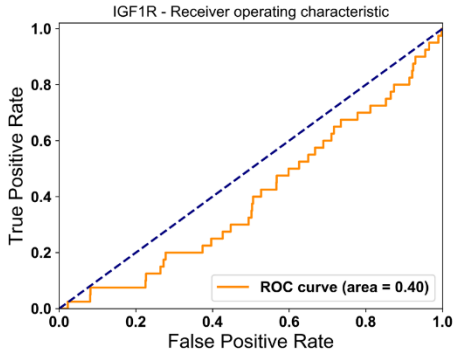

HDAC8 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

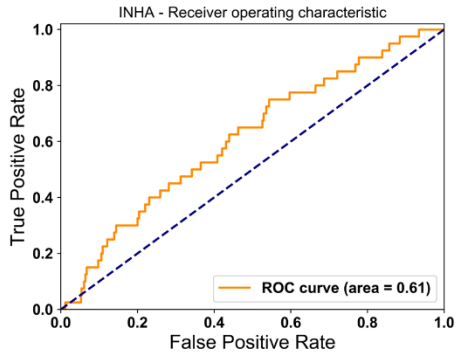

HIV1PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

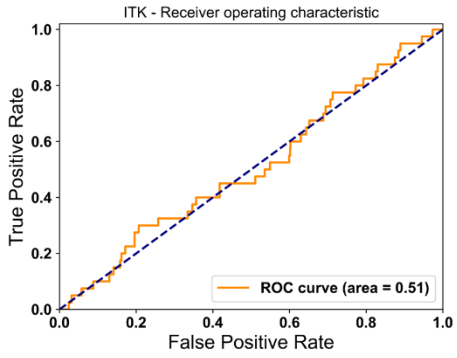

HIV1RT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.56)

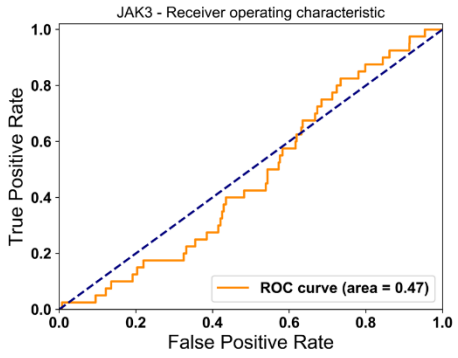

HMGR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

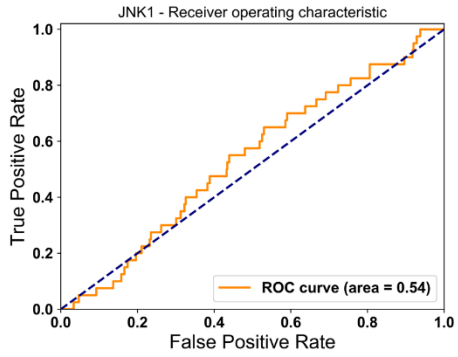

HSP90 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

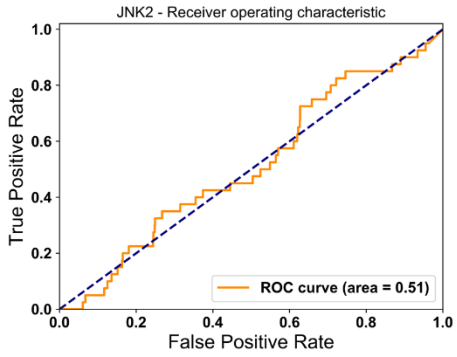

IGF1R - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.40)

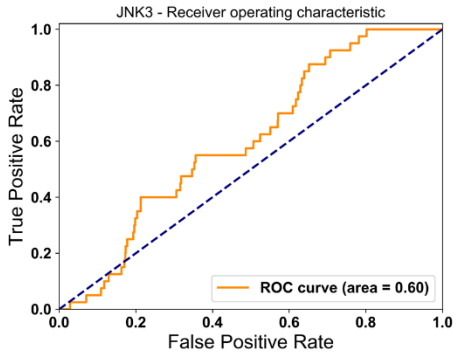

INHA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

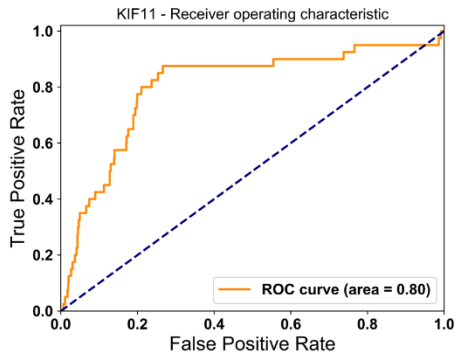

ITK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

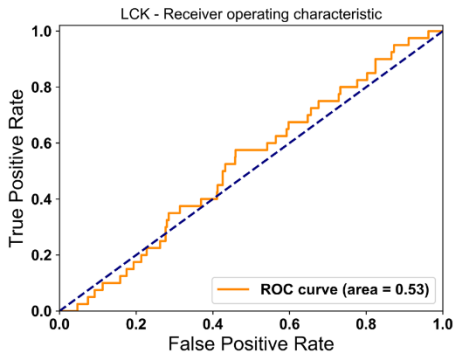

JAK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

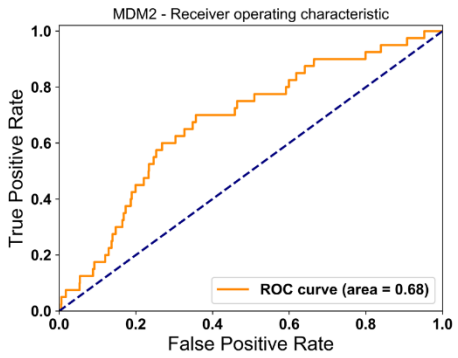

JNK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

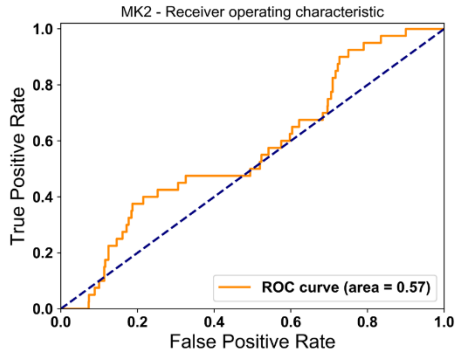

JNK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

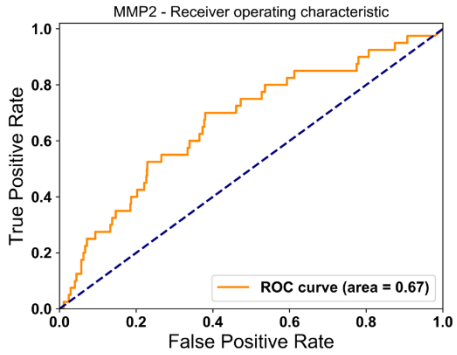

JNK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

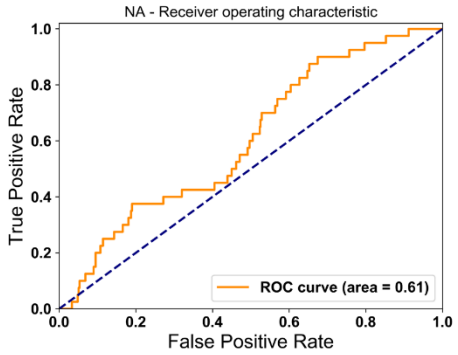

KIF11 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

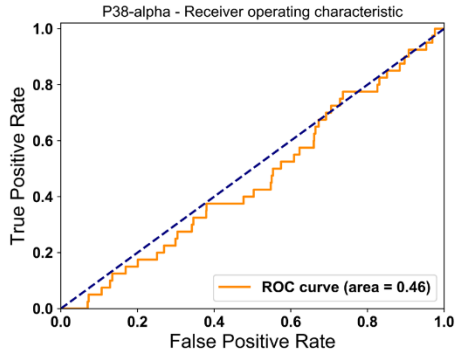

LCK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

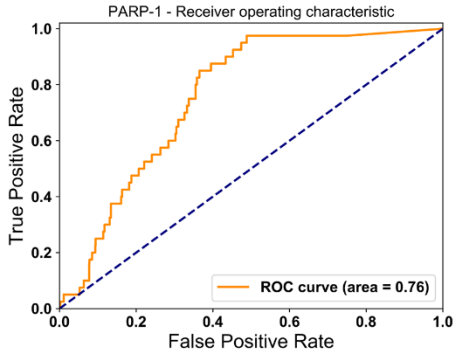

MDM2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.68)

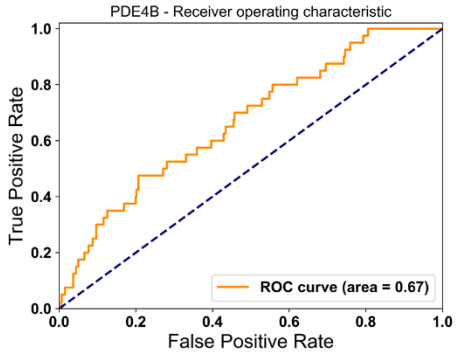

MK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

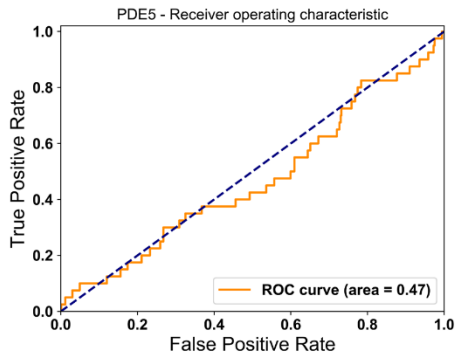

MMP2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

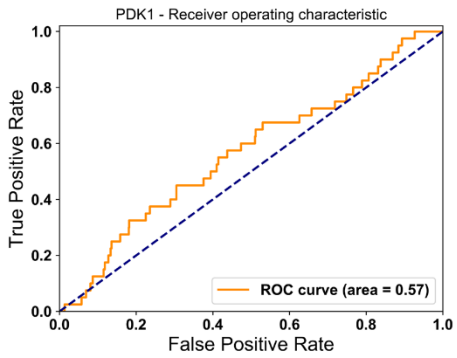

NA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

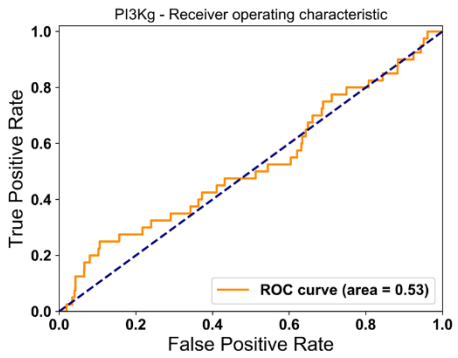

P38-alpha - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.46)

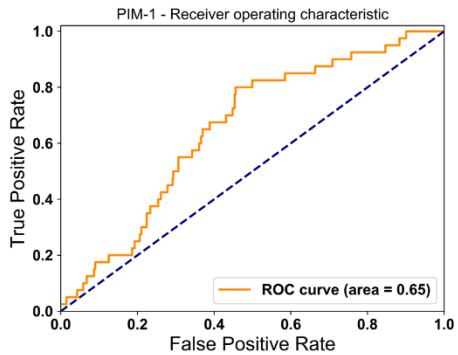

PARP-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.76)

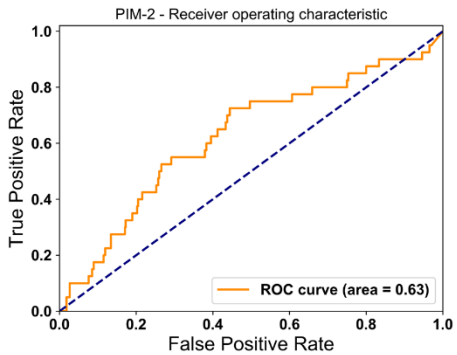

PDE4B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

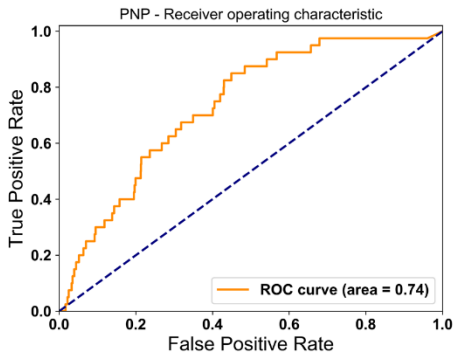

PDE5 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

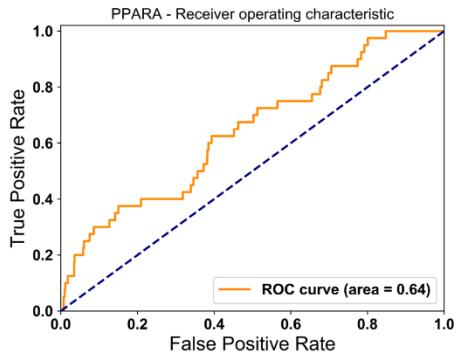

PDK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

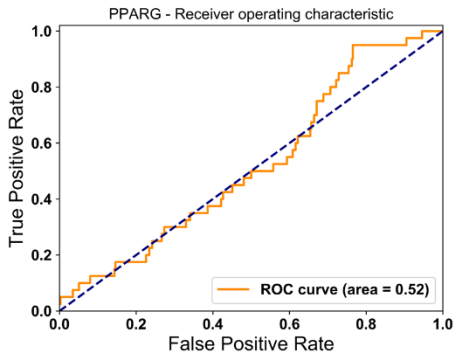

PI3Kg - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

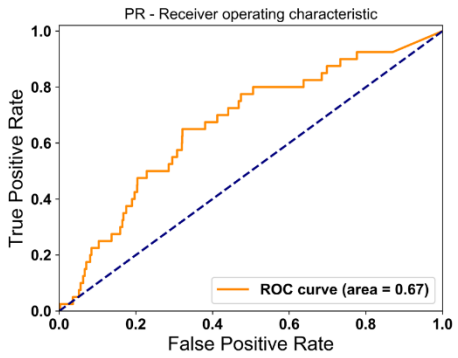

PIM-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

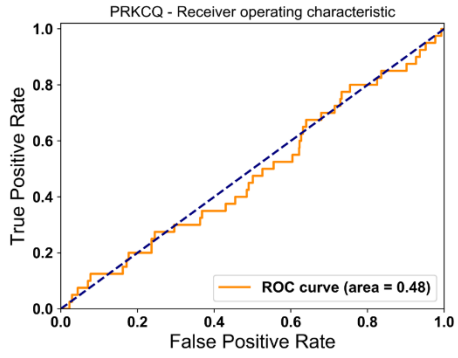

PIM-2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

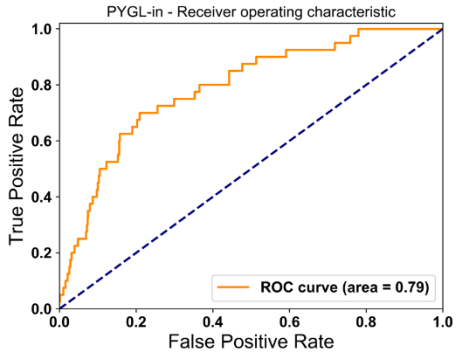

PNP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

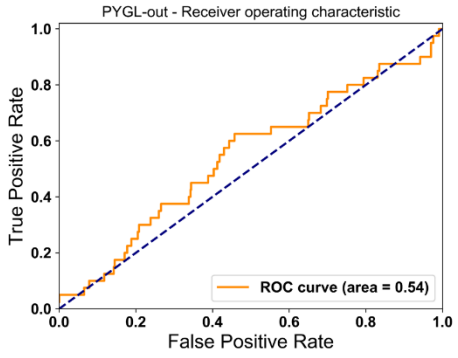

PPARA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

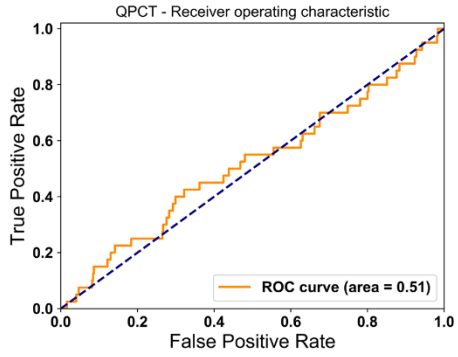

PPARG - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

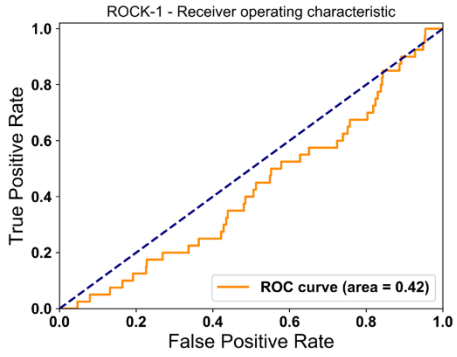

PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

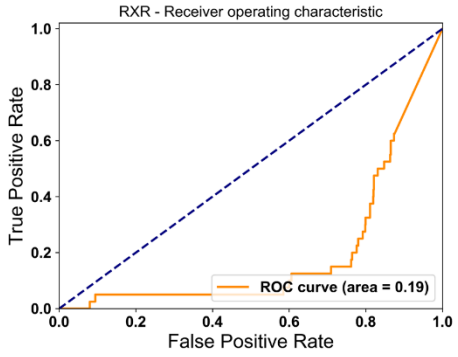

PRKCQ - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.48)

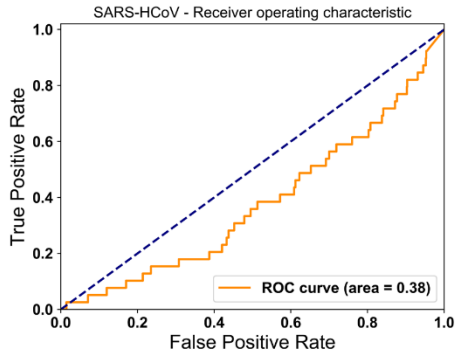

PYGL-in - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

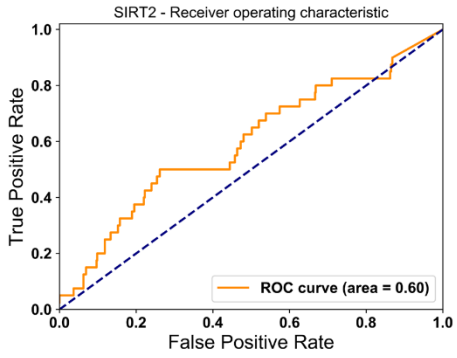

PYGL-out - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.54)

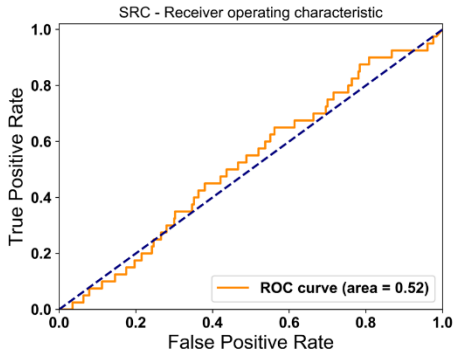

QPCT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.51)

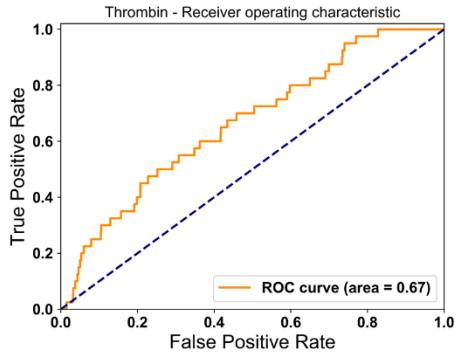

ROCK-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.42)

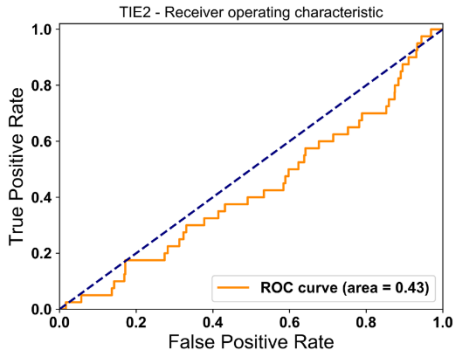

RXR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.19)

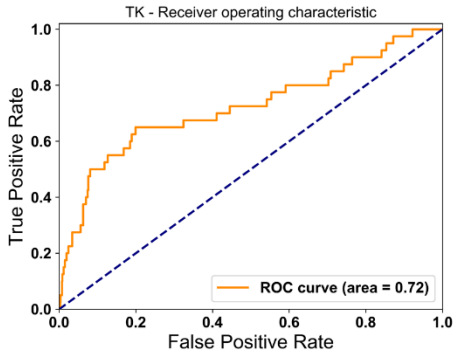

SARS-HCoV - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.38)

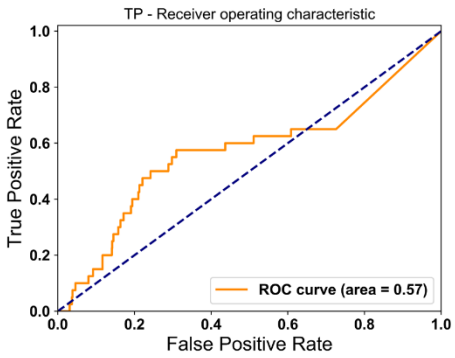

SIRT2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

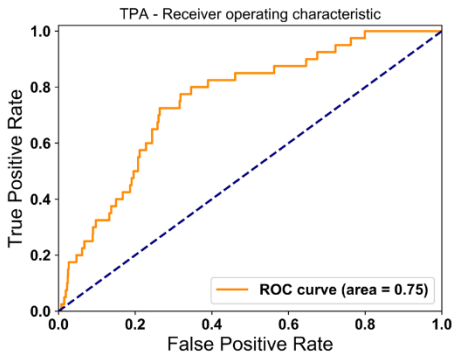

SRC - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

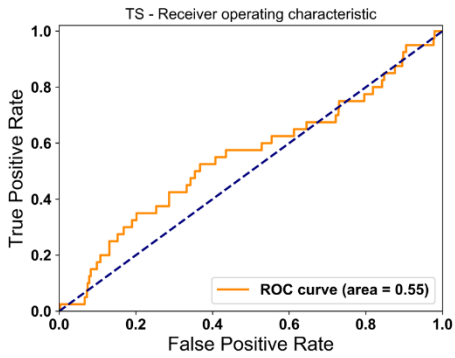

Thrombin - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

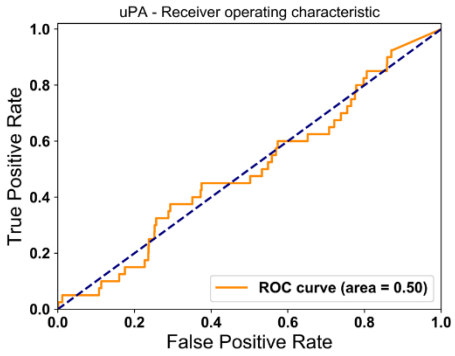

TIE2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.43)

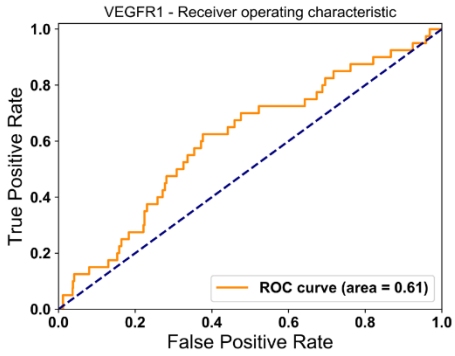

TK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

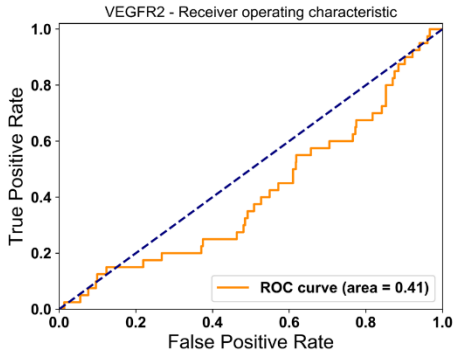

TP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)


TPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)

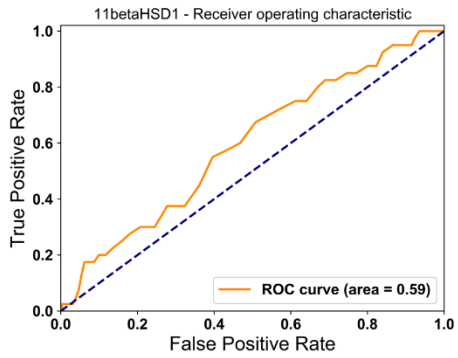

TS - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

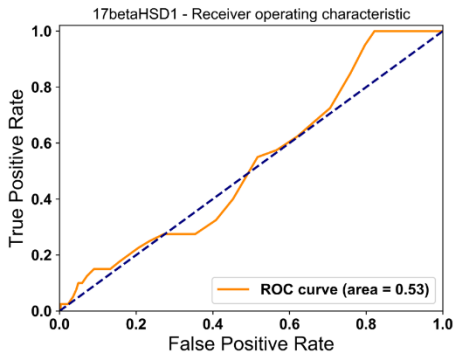

uPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.50)

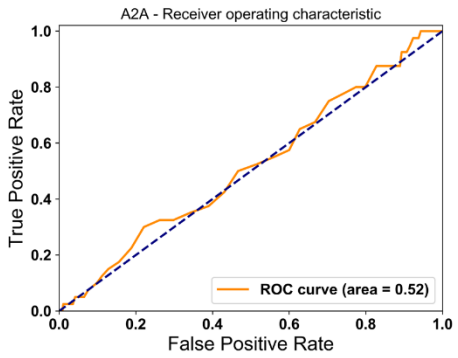

VEGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

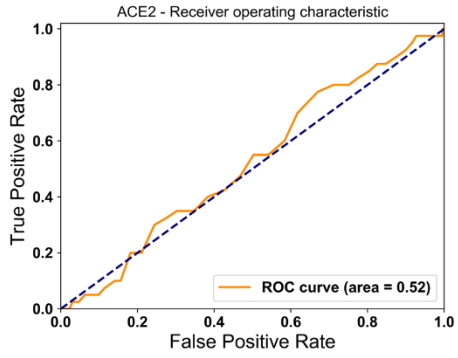

VEGFR2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.41)

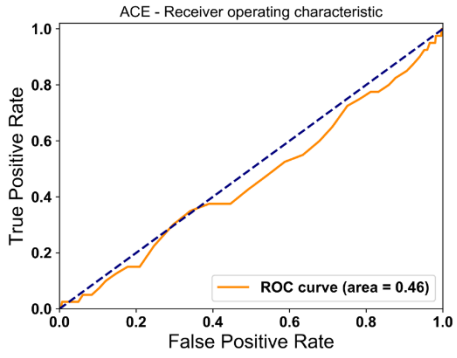
ROC of Vina flexible-docking results

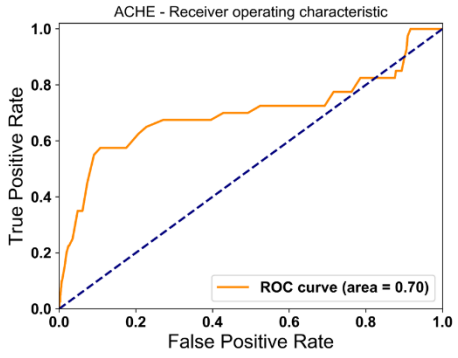


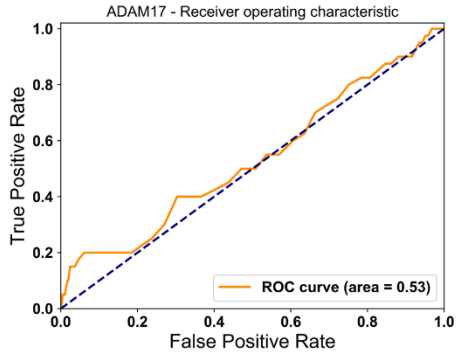


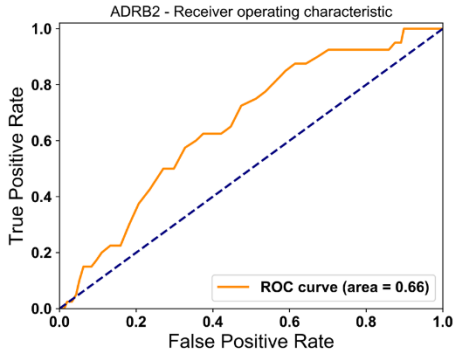


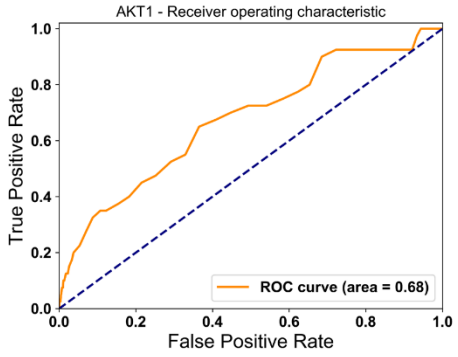


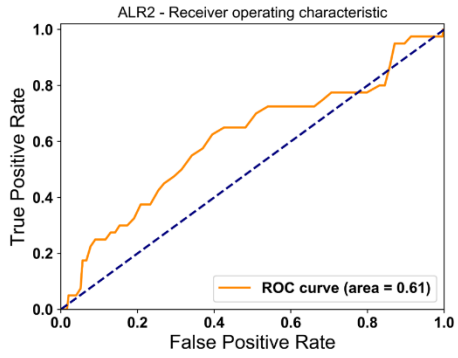


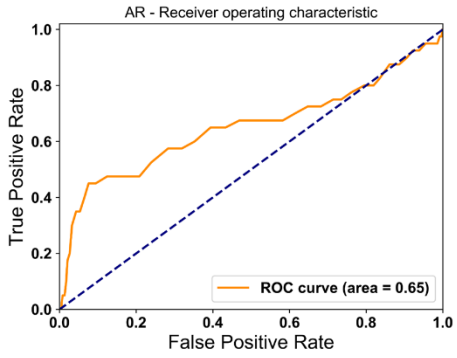


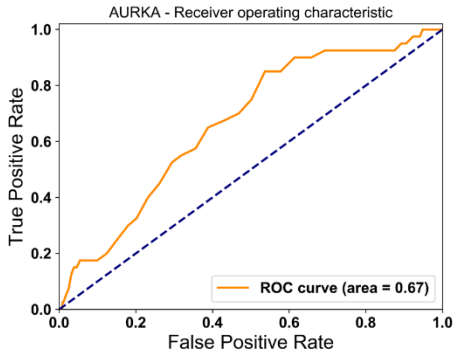


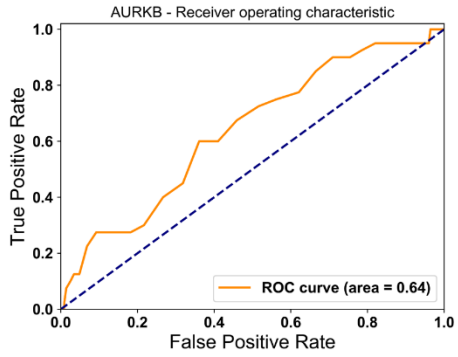


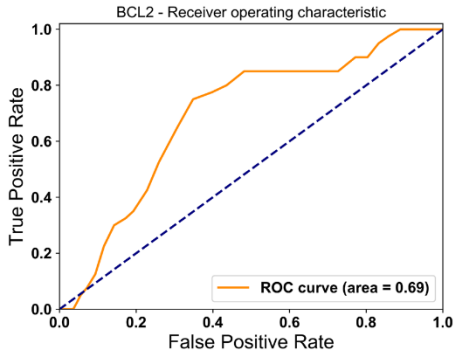


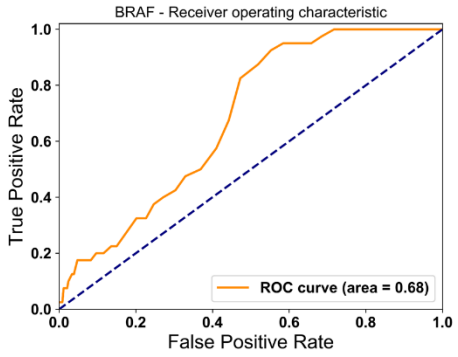


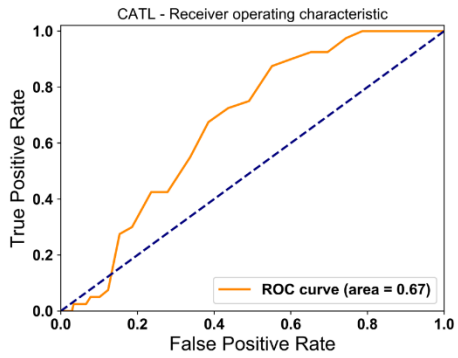


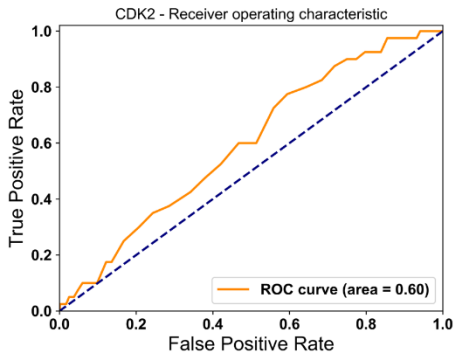


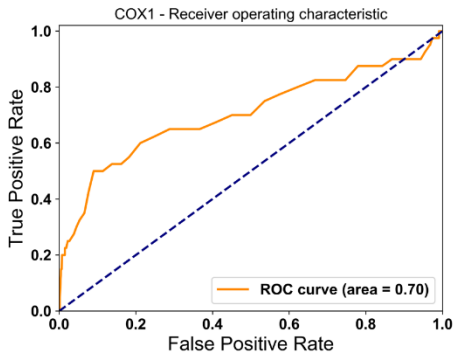


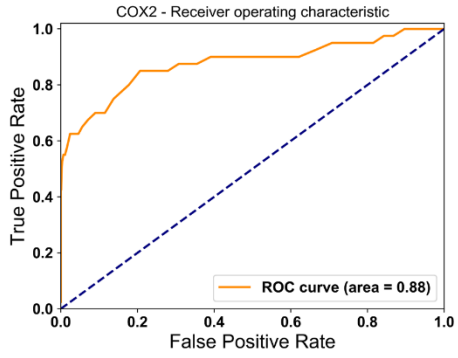


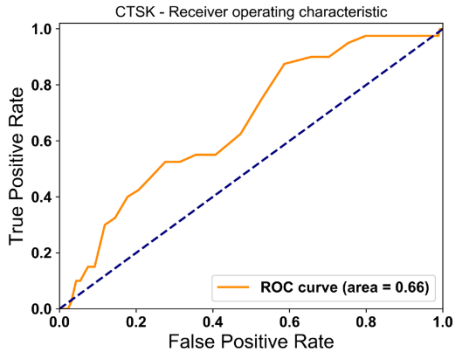


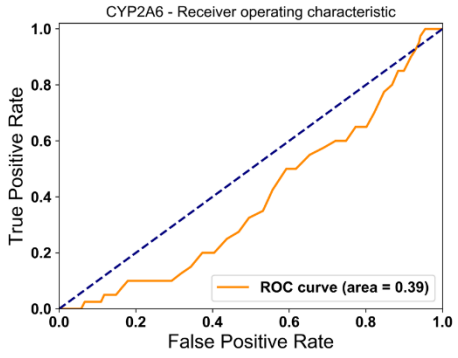

CATL - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

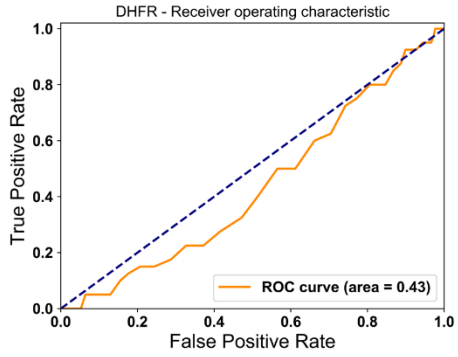

CDK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.60)

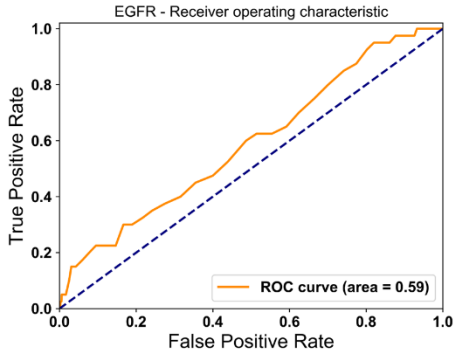

COX1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.70)

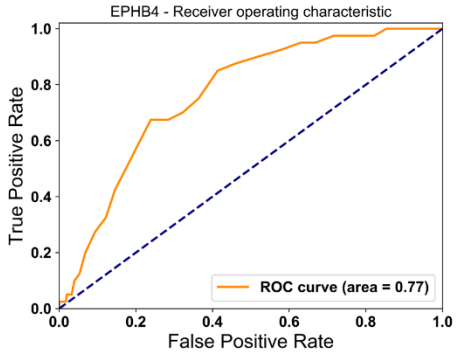

COX2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.88)

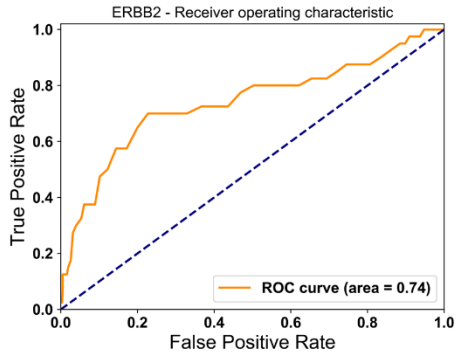

CTSK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

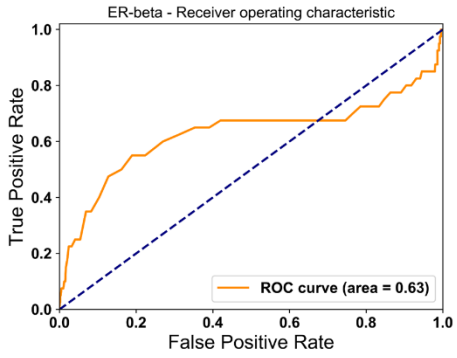

CYP2A6 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.39)

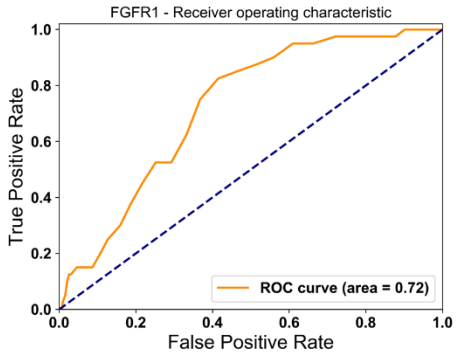

DHFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.43)

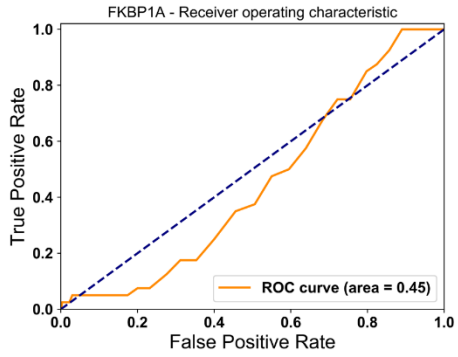

EGFR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

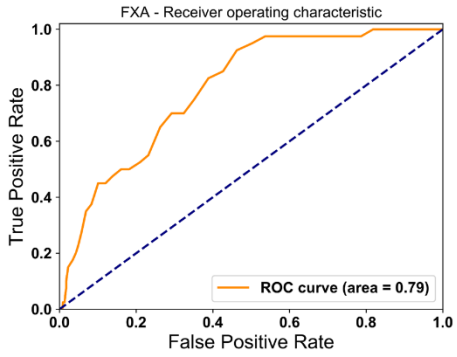

EPHB4 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.77)

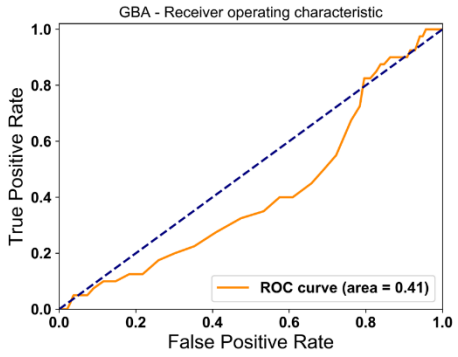

ERBB2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

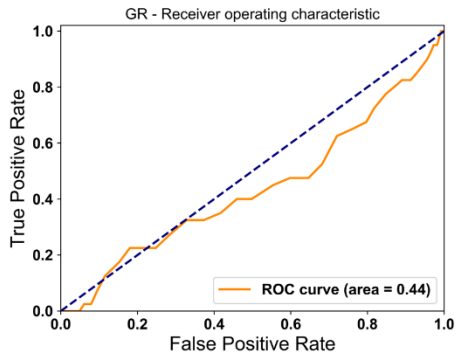

ER-beta - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

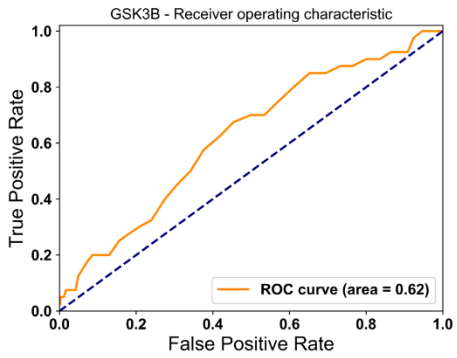

FGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

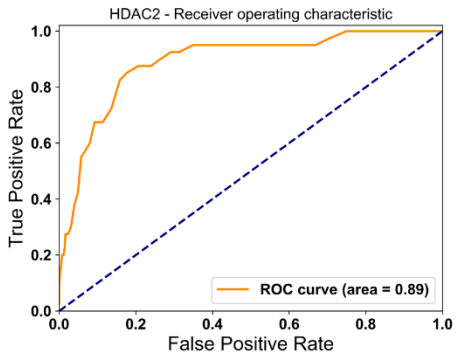

FKBP1A - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

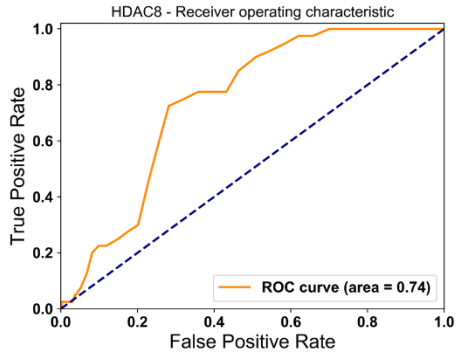

FXA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

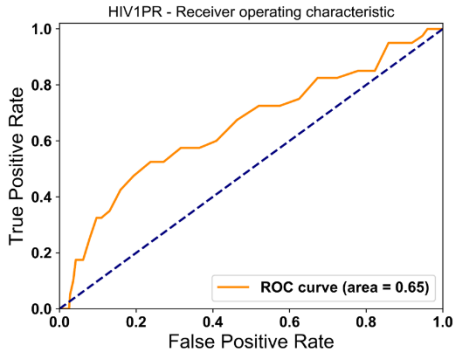

GBA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.41)

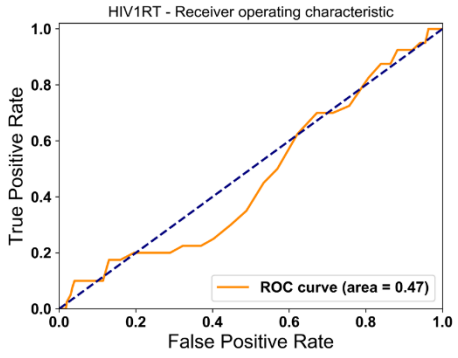

GR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.44)

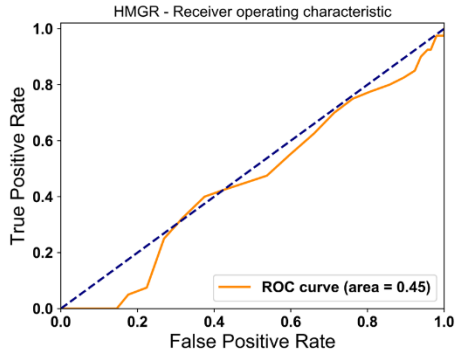

GSK3B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

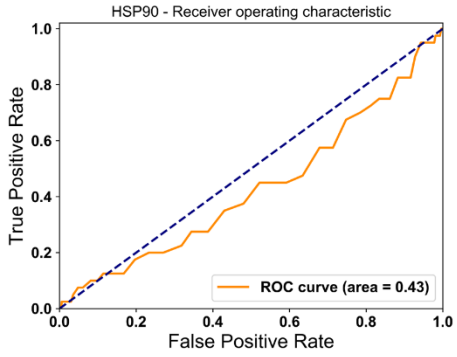

HDAC2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)

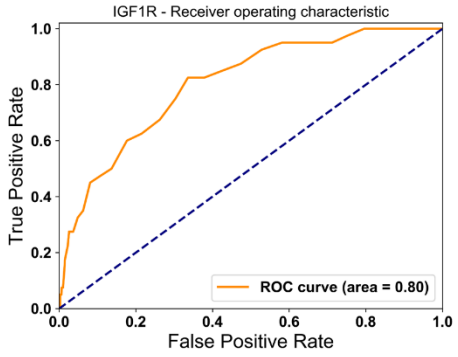

HDAC8 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

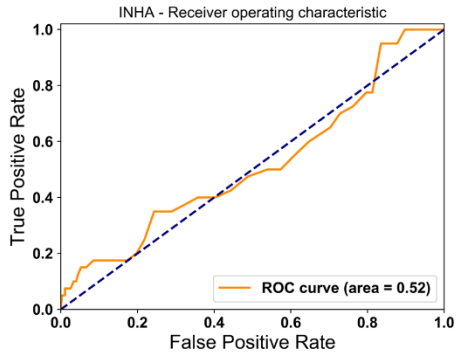

HIV1PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

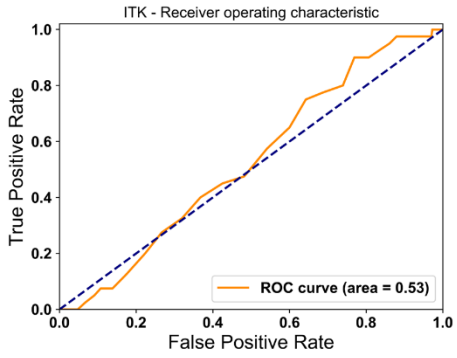

HIV1RT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.47)

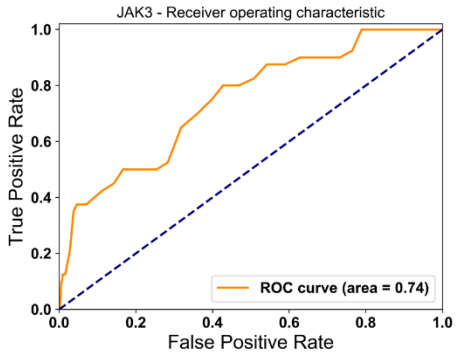

HMGR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

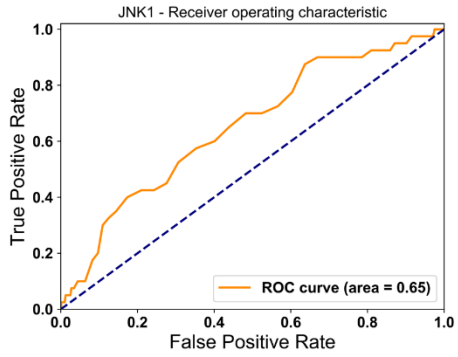

HSP90 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.43)

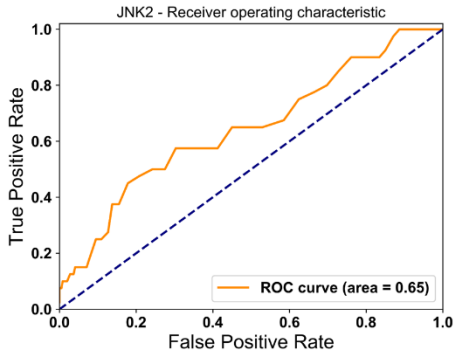

IGF1R - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

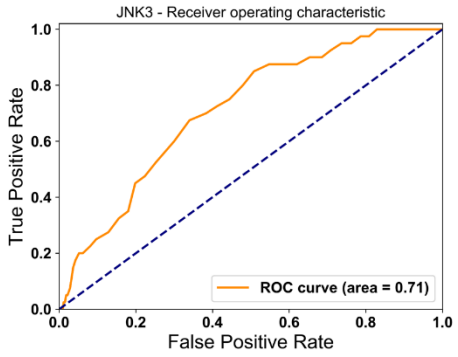

INHA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

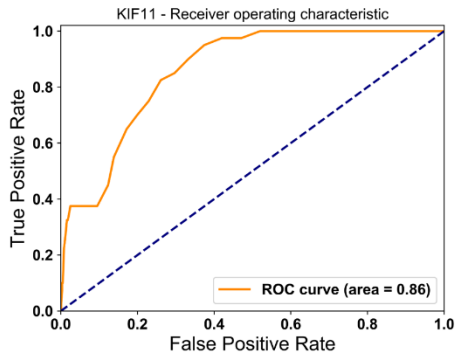

ITK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

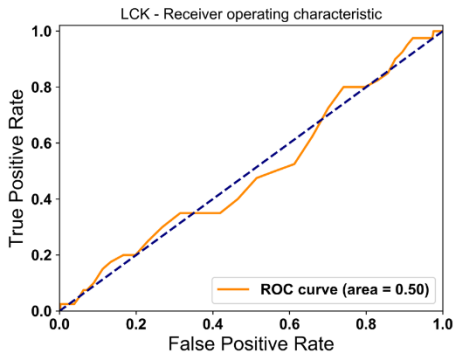

JAK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

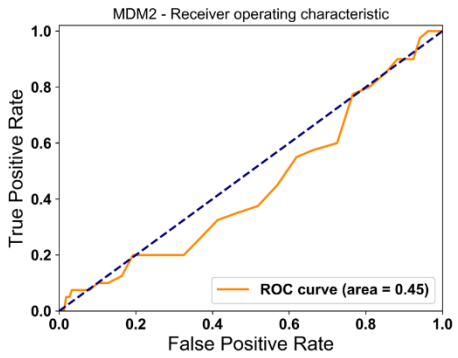

JNK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

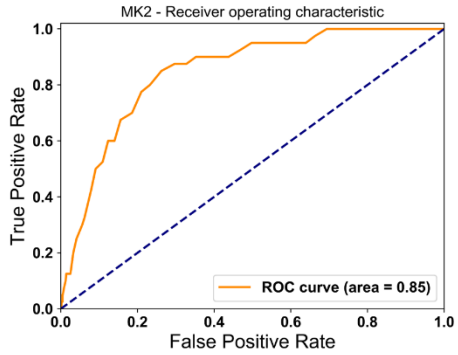

JNK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.65)

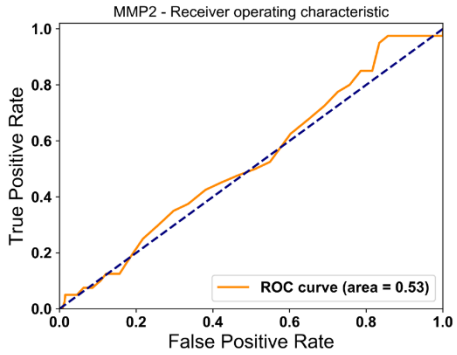

JNK3 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

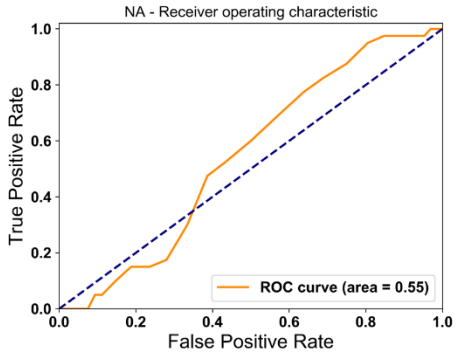

KIF11 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)

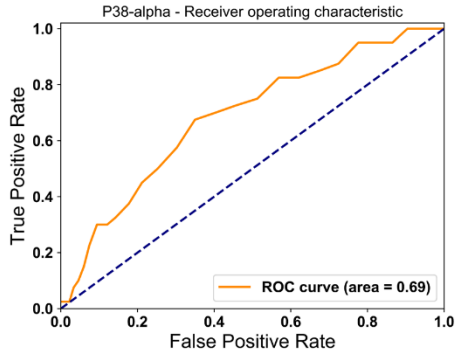

LCK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.50)

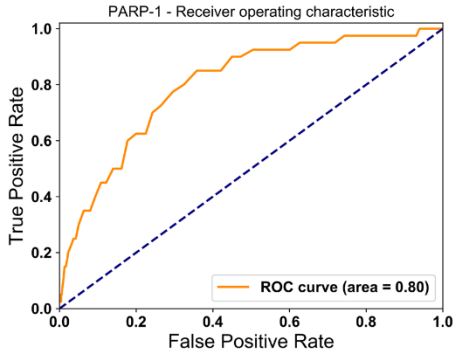

MDM2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)

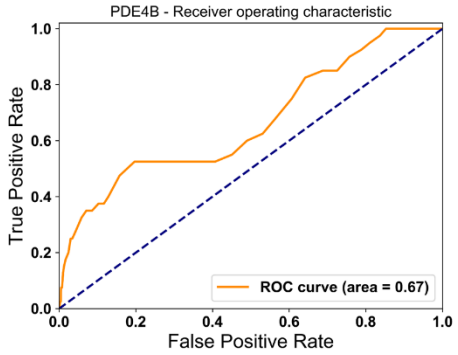

MK2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.85)

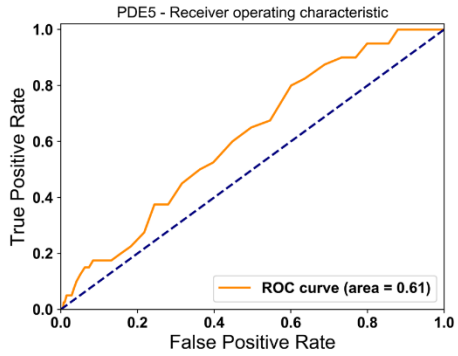

MMP2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

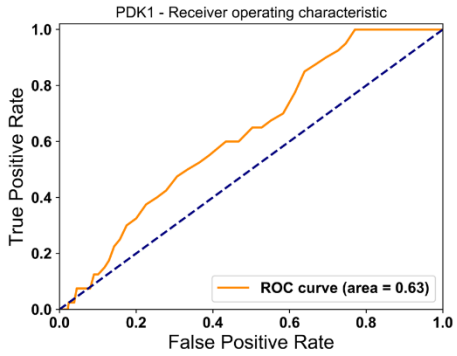

NA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.55)

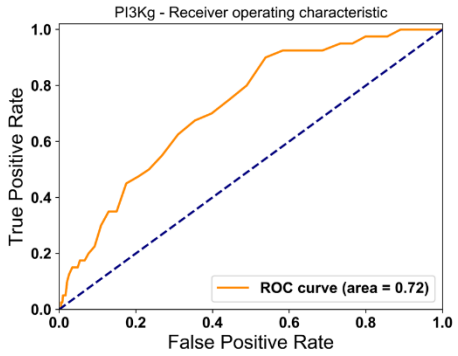

P38-alpha - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

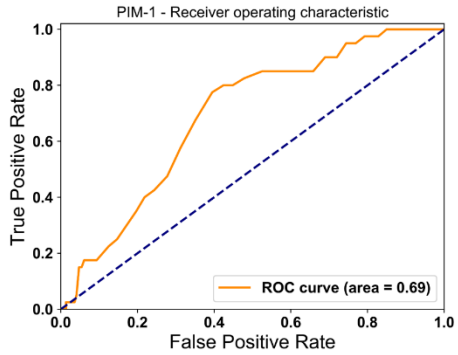

PARP-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.80)

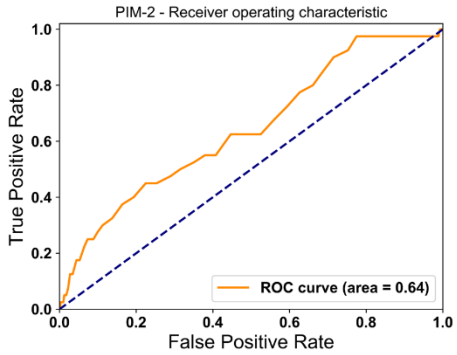

PDE4B - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)

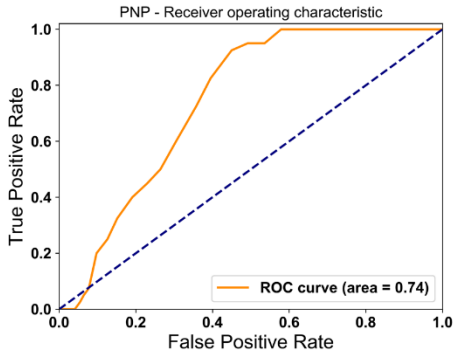

PDE5 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.61)

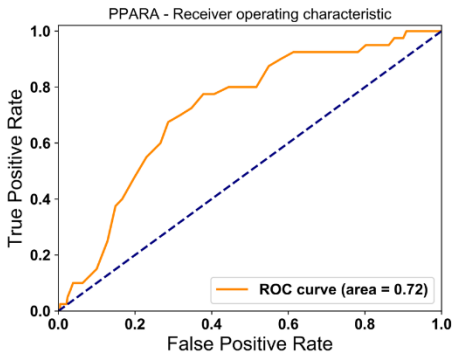

PDK1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

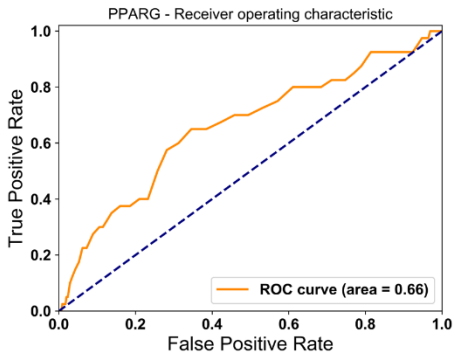

PI3Kg - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

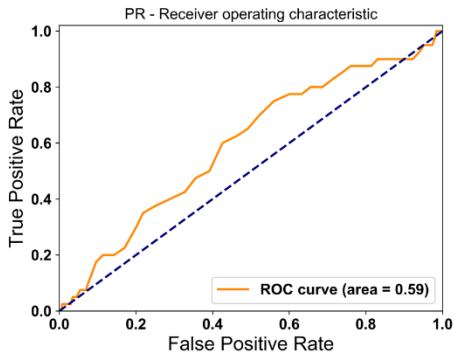

PIM-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

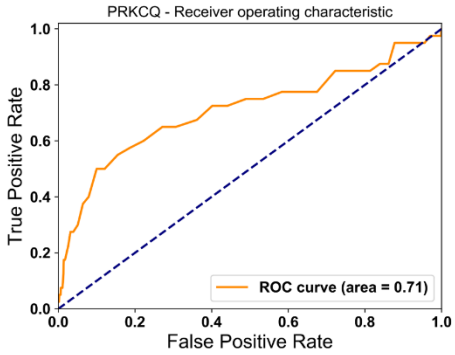

PIM-2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.64)

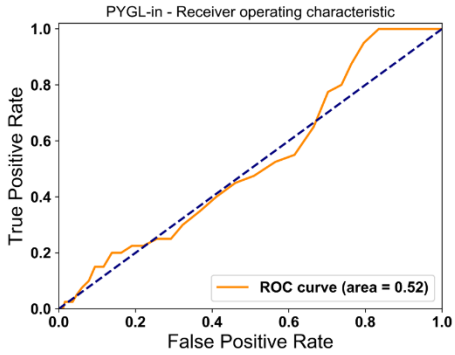

PNP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.74)

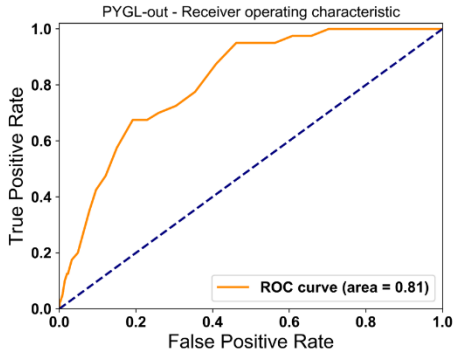

PPARA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.72)

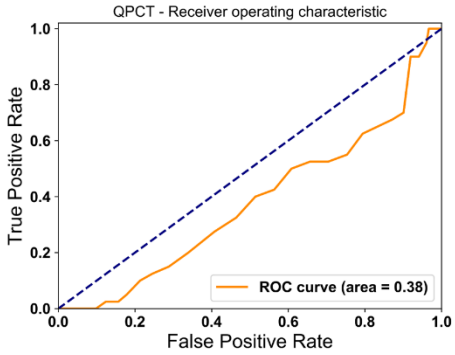

PPARG - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.66)

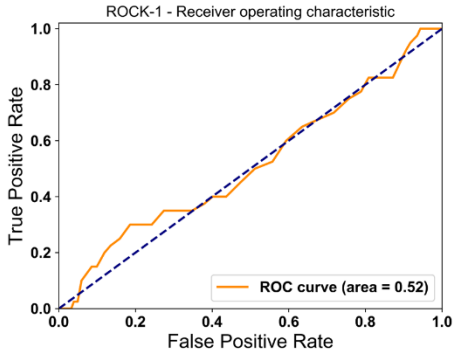

PR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.59)

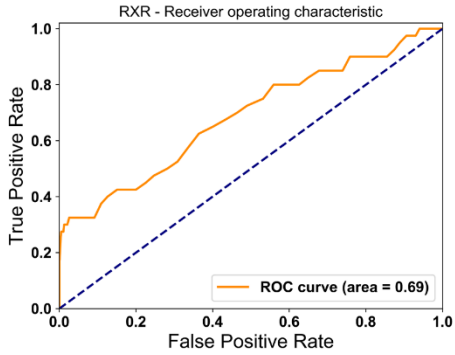

PRKCQ - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.71)

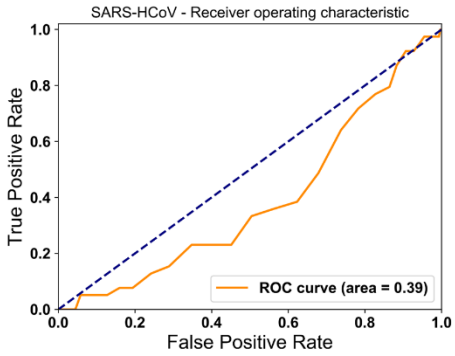

PYGL-in - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

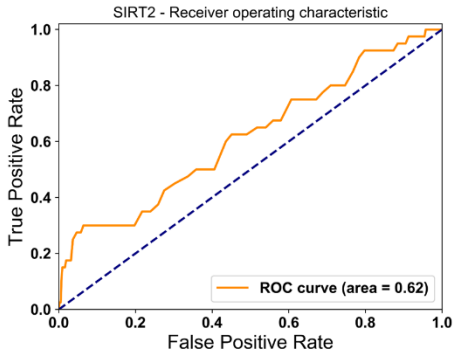

PYGL-out - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.81)

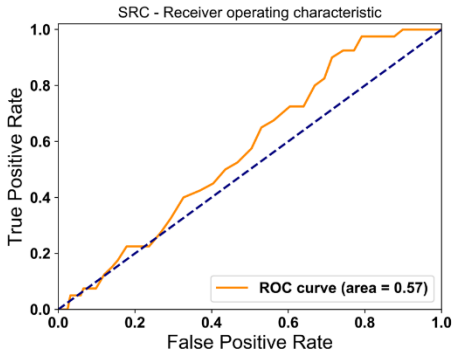

QPCT - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.38)

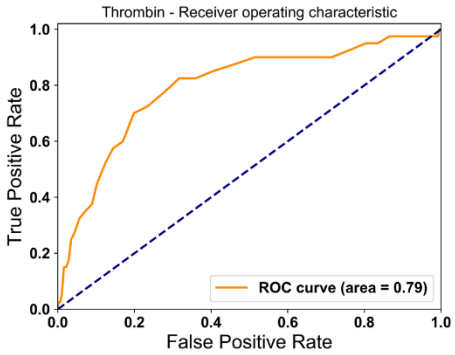

ROCK-1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.52)

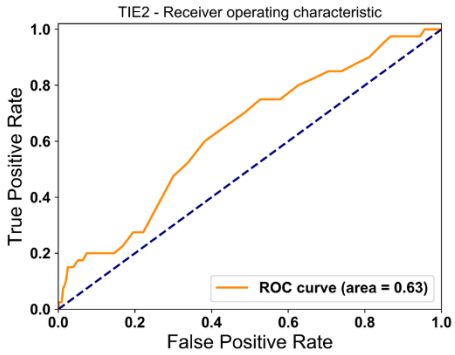

RXR - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.69)

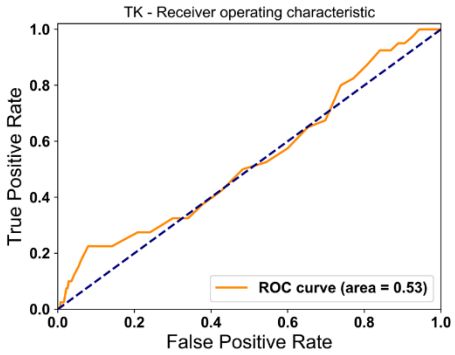

SARS-HCoV - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.39)

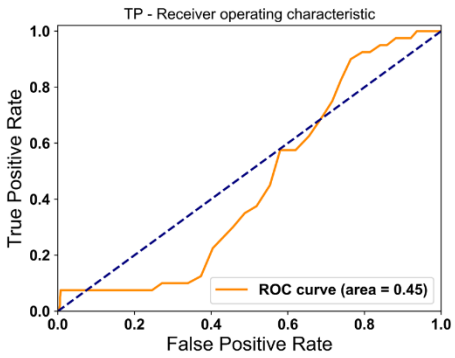

SIRT2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

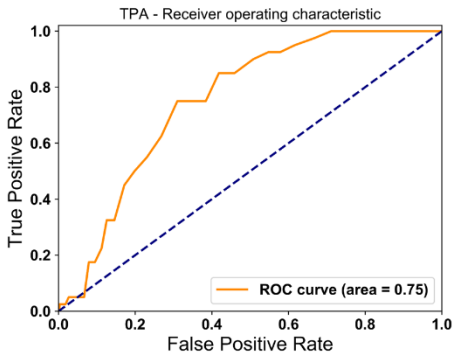

SRC - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)

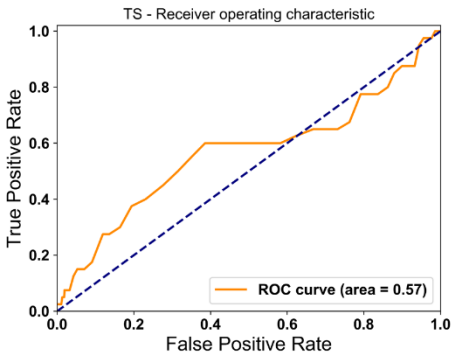

Thrombin - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.79)

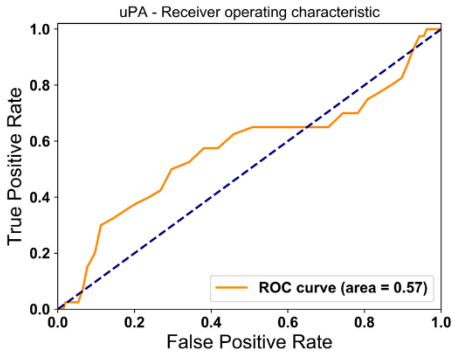

TIE2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.63)

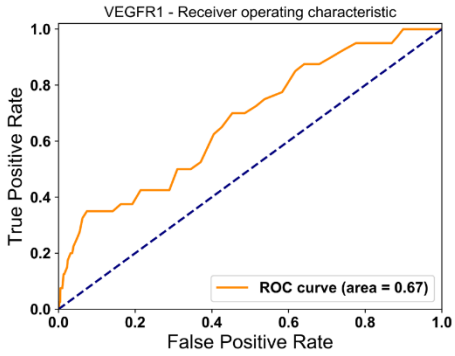

TK - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.53)

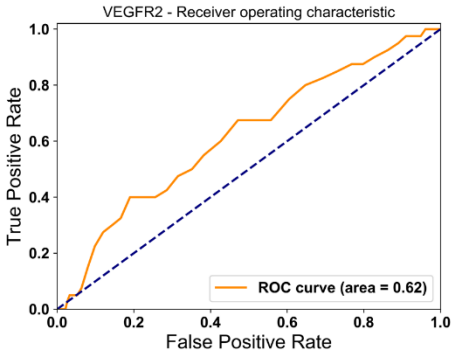

TP - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.45)


TPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.75)


TS - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)


uPA - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.57)


VEGFR1 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.67)


VEGFR2 - Receiver operating characteristic
True Positive Rate
False Positive Rate
ROC curve (area = 0.62)

Appendix 7. Code of AUC-ROC calculation

```
def calculate_auc_roc(ranking):
    """
    fp denotes false positive
    tp denotes true positive
    """
    # save the each ligand's property (active or decoy)
    # 1 denotes active; 0 denotes decoy
    y = []
    for rank in ranking:
        y.append(rank[1])
    y = tuple(y)
    tpr = []
    fpr = []
    sfp = 0.0
    p = 0.0
    # check total actives in the list
    ap = y.count(1)
    # check total decoys in the list
    an = len(y) - ap
    tp = 0.0
    fp = 0.0
    """
    if the scores of two points are different,
    the resulting area is calculated in a rectangle way
    if the scores of a series points are identical,
    the function will find the the starting point
    and the end point of such series
    then the resulting area is calculated in a trapezoid way
    """
    for j in range(len(ranking)):
        if j != len(ranking) - 1 and ranking[j][0] == ranking[j +
1][0]:
            if ranking[j][1] == 0:
                sfp = sfp + 1
            true_positive_rate = tp / ap
            false_positive_rate = fp / an
            tpr.append(true_positive_rate)
            fpr.append(false_positive_rate)
        else:
            p = j + 1
            if ranking[j][1] == 0:
                fp = fp + 1
                sfp = sfp + 1
            fp = sfp
            tp = p - fp
            true_positive_rate = tp / ap
            false_positive_rate = fp / an
            tpr.append(true_positive_rate)
            fpr.append(false_positive_rate)
    tpr = tuple(tpr)
    fpr = tuple(fpr)
    area = [[0.0, 0.0]]
    for i in range(len(fpr)):
        area.append(tuple([fpr[i], tpr[i]]))
    area = tuple(area)
    auc = 0.0
```

```
    start_point = 0.0
    end_point = 0.0
    height = 0.0
    for key in area:
        if start_point != key[0] and height != key[1]:
            end_point = key[0]
            temp = ((end_point - start_point) * (height + key[1])) /
2
            auc = auc + temp
            start_point = key[0]
            height = key[1]
        elif start_point != key[0] or height != key[1]:
            end_point = key[0]
            temp = (end_point - start_point) * height
            auc = auc + temp
            start_point = key[0]
            height = key[1]
    return auc
```

Appendix 8. Code of Entropy calculation

```
#!/usr/bin/python
# import necessary libraies;
# calculation for multiple targets in implemented with
multiprocessing
import os
import csv
import collections
import multiprocessing as mp
import time
import math

# create basic class for each atom
class Atom(object):
    def __init__(self, atom_type, coordinate):
        self.__type = atom_type
        self.__coordinate = Point(coordinate)

    def get_coordinate(self):
        return self.__coordinate

    def get_type(self):
        return self.__type


class Point(object):
    def __init__(self, coordinate):
        self.x = coordinate[0]
        self.y = coordinate[1]
        self.z = coordinate[2]
    def cal_dist(self, point):
        __distance = math.sqrt((self.x - point.x) ** 2 + (self.y -
point.y) ** 2 + (self.z - point.z) ** 2)
        return __distance
"""
the number of water that each atom can carry is according to
the number of sp hydrogens that a donor atom can donate
or the number of hydrogen bonds that an acceptor atom can accept
"""
# create atom class of residue
# when creating, the number of water carried is calculated
class ResidueAtom(Atom):
    def __init__(self, atom_type, coordinate, index, residue):
        # the number of water carried by each atom is set
        __residue_number_of_water = {"ARG": {"NE": 1, "NH1": 2,
"NH2": 2},
                                     "ASN": {"ND2": 2, "OD1":
2},
                                     "ASP": {"OD1": 2, "OD2":
2},
                                     "GLN": {"NE2": 2, "OE1":
2},
                                     "GLU": {"OE1": 2, "OE2":
2},
                                     "HIS": {"ND1": 1, "NE2":
1},
                                     "LYS": {"NZ": 3},
```

```
                                                "SER": {"OG": 2},
                                                "THR": {"OG1": 2},
                                                "TRP": {"NE1": 1},
                                                "TYR": {"OH": 1}}
        Atom.__init__(self, atom_type, coordinate)
        self.__residue = residue
        self.__index = index
        try:
            self.__number_of_water =
__residue_number_of_water[residue][atom_type]
        except KeyError:
            self.__number_of_water = 0

    def get_residue(self):
        return self.__index, self.__residue

    def get_water_count(self):
        return self.__number_of_water

# create atom class of ligand;
# when creating, the number of water carried is calculated
class LigandAtom(Atom):
    def __init__(self, atom_type, coordinate):
        # the number of water carried by each atom is set
        __ligand_number_of_water = {"N.4": 1, "N.3": 2, "N.2": 3,
"N.1": 4,                                   "N.ar": 1, "N.pl3": 1,
"N.am": 1, "O.3": 1, "O.2": 2, "O.co2": 2}
        Atom.__init__(self, atom_type, coordinate)
        try:
            self.__number_of_water =
__ligand_number_of_water[atom_type]
        except KeyError:
            self.__number_of_water = 0

    def get_water_count(self):
        return self.__number_of_water
# Side chain conformational entropy (cal/mol-1, 300K)
__Amino_acid_conformational_entropy = {"LYS": -189, "ARG": -188,
                                                "GLN": 173, "MET": -
146,
                                                "GLU": -146, "ILE": -
76,
                                                "LEU": -71, "ASN": -
103,
                                                "THR": -108, "VAL": -
43,
                                                "TYR": -113, "SER": -
111,
                                                "HIS": -95, "ASP": -
78,
                                                "CYS": -85, "TRP": -
99,
                                                "PHE": -62, "ALA": 0,
                                                "PRO": -6, "GLY": 0}

# read protein atoms and create class ResidueAtom
# for each atom
def get_pdb_atoms(pdbfile):
    atoms = []
```

```
    with open(pdbfile, 'r') as pdb:
        lines = tuple(atom.strip() for atom in pdb.readlines()[0: -
2])
    for atom in lines:
        identifier = atom[0: 6].strip()
        if identifier == 'ATOM':
            atom_type = atom[12: 16].strip()
            residue = atom[17: 20].strip()
            index = atom[22: 26].strip()
            atoms.append(ResidueAtom(atom_type=atom_type,
                                     coordinate=tuple(map(float,
(atom[30: 38].strip(), atom[38: 46].strip(), atom[46:
54].strip()))),
                                     index=index, residue=residue))
    return tuple(atoms)

# read ligand atoms and create class LigandAtom
# for each atom
def get_sdf_atoms(sdffile, single=True):
    atom_types = []
    atoms = []
    if single:
        with open(sdffile, 'r') as sdf:
            lines = tuple(sdf.readlines())
    else:
        lines = sdffile
    split_line = lines[3].strip().split()
    natoms = int(split_line[0])
    nbonds = int(split_line[1])
    if natoms > len(lines):
        count = len(split_line[0])
        while natoms > len(lines):
            natoms = int(split_line[0][:count])
            count -= 1
        nbonds = int(split_line[0][count + 1:])
    atom_lines = lines[4: 4 + natoms]
    property_lines = lines[5 + natoms + nbonds:]
    for i, line in enumerate(property_lines):
        index = i
        while line == ">  <TYPE_INFO>\n":
            next_line = property_lines[index + 1]
            if next_line == "\n":
                break
            atom_types.extend(next_line.strip().split())
            index += 1
    for i, atom in enumerate(atom_lines):
        split_atom = atom.strip().split()
        atom_type = atom_types[i]
        coordinate = tuple(map(float, split_atom[0: 3]))
        atoms.append(LigandAtom(atom_type=atom_type,
coordinate=coordinate))
    return tuple(atoms)

# function of lost water entropy and side chain conformational
entropy calculation
def cal_lw_fs(ligand, protein):
    ligand_protein_contacts = {}
    total_flexible_entropy = 0
    total_protein_water_loss = 0
```

```
    total_ligand_water_loss = 0
    deduct_water = 1
    protein_contacted_water = {}
    ligand_contacted_water = {}
    contacted_atom = set()
    # obtain contacted atoms in protein
    # contacted atom distance is set to 3.5
    for ligand_atom in ligand:
        dists = set()
        ligand_atom_coordinate = ligand_atom.get_coordinate()
        for residue_atom in protein:
            residue_atom_coordinate = residue_atom.get_coordinate()
            dist =
ligand_atom_coordinate.cal_dist(residue_atom_coordinate)
            if dist > 3.5:
                pass
            else:
                dists.add(residue_atom)
        ligand_protein_contacts[ligand_atom] = dists

    for latm in ligand_protein_contacts:
        # calculate ligand water loss
        if len(ligand_protein_contacts[latm]) >=
latm.get_water_count():
            ligand_water_loss = latm.get_water_count()
            total_ligand_water_loss += ligand_water_loss
        else:
            total_ligand_water_loss +=
len(ligand_protein_contacts[latm])

        for atm in ligand_protein_contacts[latm]:
            # calculate flexibility entropy
            flexible_entropy = 0
            index, residue = atm.get_residue()
            if index not in contacted_atom:
                contacted_atom.add(index)
                flexible_entropy =
__Amino_acid_conformational_entropy[residue]
            total_flexible_entropy += flexible_entropy
            # calculate protein water loss
            if atm not in protein_contacted_water:
                protein_contacted_water[atm] = atm.get_water_count()
            original_water = protein_contacted_water[atm]
            if original_water == 0:
                pass
            else:
                left_water = original_water - deduct_water
                total_protein_water_loss += deduct_water
                protein_contacted_water.update({atm: left_water})
    total_water_loss = total_ligand_water_loss +
total_protein_water_loss
    water_entropy = total_water_loss * 10 * 300 * 0.001
   return water_entropy, total_flexible_entropy

# for one sub-process, successively calculate the entropy
def multi_cal_entropy(job, patms):
    entropies = []
    for mol in job:
        try:
```

```
            latms = get_sdf_atoms(mol[1], single=False)
            entropies.append((mol[0], cal_lw_fs(latms, patms)))
        except:
            print(mol[0])
    return tuple(entropies)

# collect the result
def multi_collect(entropies):
    global Multi_results
    Multi_results.extend(entropies)

# split ligand job into pieces
def split_jobs(jobs, part=3):
    ratio = float(1) / part
    job = []
    for each_part in range(0, part):
        job.append(jobs[int(math.ceil(ratio * each_part *
len(jobs))): int(math.ceil(ratio * (each_part + 1) * len(jobs)))])
    return tuple(job)

# main process
if __name__ == "__main__":
    # path of input file
    path = "path of input file"
    targets = os.listdir(path)
    # protein file
    protein = "protein file"
    # ligand file
    vs_results = "ligand file"
    # path of output file
    output = "path of output file"
    outputfiles = os.listdir(output)

    for target in targets:
        # skip completed job
        if target + ".csv" in outputfiles:
            print(target + ".csv" + " done")
            pass
        else:
            Multi_results = []
            multi_sdf = []
            # read in protein atoms, create classes
            protein_atoms = get_pdb_atoms(path + target + protein)
            # read in ligand atoms, store original energy
information
            with open(path + target + vs_results, "r") as SDF:
                sdfs = SDF.readlines()
            sdffile = []
            info = []
            for i, line in enumerate(sdfs):
                sdffile.append(line)
                if line == "> <Name>\n":
                    name = sdfs[i + 1].strip()
                    info.append(name)
                if line == ">  <SCORE_INFO>\n":
                    original_energies = sdfs[i +
1].strip().split()[0: 4]
                    info.extend(original_energies)
                if line == "$$$$\n":
```

```
                    multi_sdf.append((tuple(info), sdffile))
                    sdffile = []
                    info = []
            mols = multi_sdf
            # split jobs into the number of cpu cores
            # each sub-process do calculations for multiple ligands
            start = time.time()
            p = mp.Pool(mp.cpu_count())
            jobs = split_jobs(mols, part=mp.cpu_count())
            for job in jobs:
                p.apply_async(func=multi_cal_entropy, args=(job,
protein_atoms), callback=multi_collect)
            p.close()
            p.join()
            print("%s %s" % (target, time.time() - start))
            # create output file
            field_names = ("Name", "Enthalpy", "vDW", "HBD", "HBA",
"Lost Water entropy", "Side chain conformational entropy")
            with open(output + target + ".csv", "wb") as csvfile:
                writer = csv.writer(csvfile)
                writer.writerow(field_names)
                for entropy in Multi_results:
                    writer.writerow((entropy[0][0], entropy[0][1],
entropy[0][2], entropy[0][3], entropy[0][4], entropy[1][0],
entropy[1][1] * 0.01))
```

Appendix 9. Code of downloading the actives' co-crystallized structures

```
"""
import necessary library
this program is to simulate the manual operation
of finding ligand information in BindingDB
if there is co-crystallized structure of the ligand
and the target, the program will download the PDB file
"""
from bs4 import BeautifulSoup
import time
import pandas as pd
import os
import urllib.request
from selenium import webdriver
from selenium.webdriver.support.ui import Select
from selenium.webdriver.chrome.options import Options
# path for input file
path = 'Z:\\project\\Targets'
dirs = os.listdir(path)
# set up browser
chrome_options = Options()
chrome_options.add_argument('--headless')
chrome_options.add_argument('--incognito')
# set up simulating browser
drive_path = 'C:\\Program Files
(x86)\\Google\\Chrome\\Application\\chromedriver.exe'
browser = webdriver.Chrome(drive_path,
chrome_options=chrome_options)
firsttime = True
for folder in dirs:
    print('-----------------------------------------------------------
-------------------------------------------------')
    print(folder + ' STARTED')
    file = path + '\\' + folder + '\\' + 'ligand_pandas.csv'
    actives = pd.read_csv(file)
    if 'Target' not in actives:
        print(folder + ' No TARGET')
        pass
    else:
        for i in range(len(actives)):
            name = actives['Name'][i]
            realname = name[0: 3] + 'M' + name[3:]
            targetname = actives['Target'][i]
            browser.get(url='https://www.bindingdb.org/bind/as.jsp')
            if firsttime:
                firsttime = False
            else:
                for j in range(2):
                    clear =
browser.find_element_by_class_name('clrANbtn')
                    clear.click()
            options = Select(browser.find_element_by_id('criteria'))
            o1 = options.select_by_value('target_name')
            target = browser.find_element_by_name('tn_text:0')
            target.send_keys(targetname)
```

```
            checkbox =
browser.find_element_by_name('with_3d_only:0')
            checkbox.click()
            o2 = options.select_by_value('compound_name')
            compound = browser.find_element_by_name('cn_text:0')
            compound.send_keys(realname)
            submit = browser.find_element_by_id('formSubmit')
            submit.click()
            time.sleep(0.5)
            soup = BeautifulSoup(browser.page_source, "html.parser")
            if len(soup.find_all('td')) == 0:
                print(realname + ' No PDB')
            else:
                for cell in soup.find_all('td'):
                    if "3D Structure (crystal)" in cell.descendants:
                        link = cell.findChild().get('href')
                        break
                reallink = 'http://www.bindingdb.org' + link
                browser.get(reallink)
                time.sleep(1)
                subsoup = BeautifulSoup(browser.page_source,
"html.parser")
                if subsoup.find('span',
class_="red").parent.get('href') is None:
                    link =
subsoup.find_all('td')[0].findChild().get('href')
                else:
                    link = subsoup.find('span',
class_="red").parent.get('href')
                try:
                    PDBcode = link[-4:]
                    PDBaddress = 'https://files.rcsb.org/view/' +
PDBcode + '.pdb'
                    response = urllib.request.urlopen(PDBaddress)
                    result = response.read().decode('utf-8')
                    with open('Z:\\project\\Actives\\' + folder +
"\\" + PDBcode + ".pdb", "w+") as w:
                        w.write(result)
                    print(realname + ' downloaded')
                except:
                    print('----------------------------------------
'+realname+'----------------------------------------')
browser.quit()
```

Appendix 10. Code of binding site extraction

```
#!/usr/bin/python
"""
this program is just to run PyMOL in silence
load protein file, load the ligand file, mark the residues around
the ligand within 3.5 angstroms, write in an output file
"""
from pymol import cmd
import os
import time
# path of the protein file
path = 'path of the protein file'
dirs = os.listdir(path)
# path of the ligand file
path2 = 'path of the ligand file'
dirs2 = os.listdir(path2)
# path of the output file
path3 = 'path of the output file'
with open(path3, 'w+') as w:
    for dir in dirs:
        # if there are additional actives of the same target
        if dir in dirs2:
            files = os.listdir(path + dir)
            protein = path + dir + '\\' + 'protein file'
            natlig = path + dir + '\\' + 'ligand file'
            ligand = 'additional actives of the same target'
            cmd.load(protein, 'protein')
            print(dir + ' loaded')
            cmd.load(natlig, 'natlig')
            print('Natural Ligand loaded')
            cmd.load(ligand, 'ligand')
            print('Ligand loaded')
            cmd.show_as('lines', 'all')
            print('start calculating')
            command = 'byres ligand around 3.5 in protein'
            cmd.select('contacts1', command)
            command = 'byres natlig around 3.5 in protein'
            cmd.select('contacts2', command)
            cmd.select('contacts', 'contacts1 contacts2')
        else:
            files = os.listdir(path + dir)
            protein = path + dir + '\\' + 'protein file'
            ligand = path + dir + '\\' + 'ligand file'
            cmd.load(protein, 'protein')
            print(dir + ' loaded')
            cmd.load(ligand, 'ligand')
            print('Ligand loaded')
            cmd.show_as('lines', 'all')
            command = 'byres ligand around 3.5'
            print('start calculating')
            cmd.select('contacts', command)
        residue = {}
        cmd.iterate('contacts', 'residue.update({resi: resn})')
        cmd.delete('protein')
        cmd.delete('ligand')
        cmd.delete('natlig')
        for key, value in residue.items():
```

```
            w.write(dir+' '+value+'-'+key+'\n')
        print(dir + ' Completed')
        time.sleep(0.05)
```

Appendix 11. Core code of binding site geometric descriptor

```
import math
import os
# calculate three values for a set of distances
def cal_coordinate(distlist):
    mean_dist = sum(tuple(distlist.values()))/len(distlist)
    var_dist = sum(((dist - mean_dist) ** 2 for dist in
tuple(distlist.values()))) / (len(distlist) - 1)
    sd = math.sqrt(var_dist)
    skew_dist = sum(((dist - mean_dist) / sd) ** 3 for dist in
tuple(distlist.values())) / len(distlist)
    return mean_dist, var_dist, skew_dist

# calculate coordinate for all distances
def pack_coordinates(distlist1, distlist2, distlist3, distlist4):
    coordinate = []
    for dl in (distlist1, distlist2, distlist3, distlist4):
        coordinate.extend(cal_coordinate(distlist=dl))
    return tuple(coordinate)

# calculte atom distance
def cal_dist(atom, ref):
    dist = math.sqrt((atom[0] - ref[0]) ** 2 + (atom[1] - ref[1]) **
2 + (atom[2] - ref[2]) ** 2)
    return dist

# calculate center of gravity
def cal_cog(atoms):
    x = 0
    y = 0
    z = 0
    for atom in atoms:
        x = x + atom[0]
        y = y + atom[1]
        z = z + atom[2]
    cog = (x / len(atoms), y / len(atoms), z / len(atoms))
    return cog

# calculate distance from a atom to other atoms
def accu_dist(atoms, ref):
    dists = dict()
    for atom in atoms:
        dist = cal_dist(atom=atom, ref=ref)
        dists[atom] = dist
    return dists

# basic descriptor and atom type descriptor, please refer to Shave
et al. (2015)
"""
Shave, S., Blackburn, E.A., Adie, J., Houston, D.R., Auer, M.,
Webster, S.P., Taylor, P. and Walkinshaw, M.D. 2015.
UFSRAT: ultra-fast shape recognition with atom types-the discovery
of novel bioactive small molecular scaffolds for FKBP12 and 11βHSD1.
PloS one. 10(2), p.e0116570.
"""
```

```
class Descriptor(object):
    def __init__(self, atoms):
        if len(atoms) != 0:
            self.__cog = cal_cog(atoms=atoms)
            self.__cog_dists = accu_dist(atoms=atoms,
ref=self.__cog)
            self.__cog_fur = max(self.__cog_dists, key=lambda k:
k[1])
            self.__cog_near = min(self.__cog_dists, key=lambda k:
k[1])
            self.__cog_near_dists = accu_dist(atoms=atoms,
ref=self.__cog_near)
            self.__cog_fur_dists = accu_dist(atoms=atoms,
ref=self.__cog_fur)
            self.__cog_fur_fur = max(self.__cog_fur_dists,
key=lambda k: k[1])
            self.__cog_fur_fur_dists = accu_dist(atoms=atoms,
ref=self.__cog_fur_fur)
            self.__coordinate = pack_coordinates(self.__cog_dists,
self.__cog_near_dists,

self.__cog_fur_dists, self.__cog_fur_fur_dists)
        else:
            self.__cog = None
            self.__cog_near = None
            self.__cog_fur = None
            self.__cog_fur_fur = None
            self.__cog_dists = None
            self.__cog_near_dists = None
            self.__cog_fur_dists = None
            self.__cog_fur_fur_dists = None
            self.__coordinate = tuple(99 for i in range(12))

    def get_coordinate(self):
        return self.__coordinate

class AtomTypeDescriptor(object):
    def __init__(self, carbons, oxygens, nitrogens, target):
        coordinate = []
        self.__target = target
        self.__carbon = Descriptor(carbons)
        self.__oxygen = Descriptor(oxygens)
        self.__nitrogen = Descriptor(nitrogens)
        coordinate.extend(self.__carbon.get_coordinate())
        coordinate.extend(self.__oxygen.get_coordinate())
        coordinate.extend(self.__nitrogen.get_coordinate())
        self.__coordinate = tuple(coordinate)

    def get_coordinate(self):
        return self.__coordinate

    def get_target(self):
        return self.__target

def read_residues(pdbfile):
    residues = dict()
    with open(pdbfile, 'r') as pdb:
        pdb_file = tuple(atom.strip() for atom in pdb.readlines()[1:
-2])
```

```
    for atom in pdb_file:
        identifier = atom[0: 6].strip()
        if identifier == 'ATOM':
            index = atom[22: 26].strip()
            atom_type = atom[13]
            if index not in residues:
                residues[index] = []
            residues[index].append((atom_type, tuple(map(float,
(atom[30: 38].strip(), atom[38: 46].strip(), atom[46:
54].strip()))))))
    return residues


def read_aalist(file):
    aalist = dict()
    with open(file, 'r') as aa_list:
        for line in aa_list:
            words = line.split()
            if words[0] not in aalist:
                aalist[words[0]] = []
            aalist[words[0]].append(words[1])
    return aalist

# calculate similarity
def cal_similarity(targetlist):
    result = dict()
    for i in range(len(targetlist)):
        suj_similarity = dict()
        ref = targetlist[i].get_coordinate()
        for j in range(len(targetlist)):
            can = targetlist[j].get_coordinate()
            dist = math.sqrt(sum(((ref[k] - can[k]) ** 2 for k in
range(36)))) / 36
            similarity = float(1) / (1 + dist)
            suj_similarity[targetlist[j].get_target()] = similarity
        result[targetlist[i].get_target()] = suj_similarity
    return result

# A list of the components of the binding site for each target
AA_List = read_aalist('binding sites of targets')
BP_list = []
# path of the protein file
path = 'path of the protein file'
# extract the carbons, nitrogens, oxygens from each residue in the
binding site for each target
# use these atoms to create above atom type descripor
for target in AA_List:
    print(target)
    Carbons = []
    Oxygens = []
    Nitrogens = []
    Atoms = []
    files = os.listdir(path + target)
    Residues = read_residues(path + target + 'protein file')
    for KeyResidue in AA_List[target]:
        try:
            resn = KeyResidue.split('-')[1]
            for atom in Residues[resn]:
                if atom[0] == 'C':
```

```
                    Carbons.append(atom[1])
                elif atom[0] == 'O':
                    Oxygens.append(atom[1])
                elif atom[0] == 'N':
                    Nitrogens.append(atom[1])
        except KeyError:
            print(KeyResidue)
    BP = AtomTypeDescriptor(carbons=Carbons, oxygens=Oxygens,
nitrogens=Nitrogens, target=target)
    BP_list.append(BP)
Result = cal_similarity(BP_list)
```

Appendix 12. Core code of re-scoring

```
#!/usr/bin/python
import math
import os
import csv
import multiprocessing as mp
from collections import OrderedDict
from rdkit.ML.Scoring.Scoring import CalcBEDROC, CalcEnrichment

# tell active or decoy
def tell_active(name):
    if 'BDB' in name:
        return 1
    if 'ZINC' in name:
        return 0

# retruan a complete ranking list
def return_ranking(output, special=False):
    output = list(sorted(iter(output), key=lambda m: m[1],
reverse=False))
    ranking = OrderedDict()
    true_rank = []
    lds = []
    ap_count = 0
    for mol in output:
        lds.append(mol[1])
        if mol[0] in ranking:
            lds = []
        else:
            active = tell_active(mol[0])
            lds.append(active)
            if active == 1:
                ap_count = ap_count + 1
            ranking[mol[0]] = lds
            lds = []
    for key, value in ranking.items():
        true_rank.append(tuple(value))
    if special:
        left_actives = 39 - ap_count
        left_decoys = 1209 - left_actives - len(true_rank)
    else:
        left_actives = 40 - ap_count
        left_decoys = 1240 - left_actives - len(true_rank)
    if left_actives != 0:
        for i in range(left_actives):
            true_rank.append((0, 1))
    if left_decoys != 0:
        for i in range(left_decoys):
            true_rank.append((0, 0))
   return tuple(true_rank)

# function for reading file
def read_csv(csvfile, heading=True):
    if heading:
        with open(csvfile, 'r') as csv:
            lines = tuple(tuple(line.strip().split(',')) for line in
csv.readlines()[1:])
```

```
        return lines
    else:
        with open(csvfile, 'r') as csv:
            lines = tuple(tuple(line.strip().split(',')) for line in
csv.readlines())
        return lines

def create_jobs(step, reduced=True, pip=False):
    jobs = []
    ratio_set = set()
    if reduced:
        for j in range(step):
            for k in range(step):
                for l in range(step):
                    ratio = cal_ratio(a=j, b=k, c=l)
                    if ratio not in ratio_set:
                        ratio_set.add(ratio)
                        if pip:
                            for i in range(4):
                                jobs.append((j, k, l, i))
                        else:
                            jobs.append((j, k, l))
        return tuple(jobs)
    else:
        for j in range(step):
            for k in range(step):
                for l in range(step):
                        if pip:
                            for i in range(4):
                                jobs.append((j, k, l, i))
                        else:
                            jobs.append((j, k, l))
        return tuple(jobs)

# calculate the ratio of weights
def cal_ratio(a, b, c):
    a = float(a)
    b = float(b)
    c = float(c)
    if a == 0.0 and b == 0.0 and c == 0.0:
        ratio = (0.0, 0.0, 0.0)
        return ratio
    else:
        total = a + b + c
        a_ratio = a / total
        b_ratio = b / total
        c_ratio = c / total
        ratio = (a_ratio, b_ratio, c_ratio)
        return ratio

# split jobs for each sub-process
def split_jobs(jobs, part=3):
    ratio = float(1) / part
    job = []
    for each_part in range(0, part):
        job.append(jobs[int(math.ceil(ratio * each_part *
len(jobs))): int(math.ceil(ratio * (each_part + 1) * len(jobs)))])
    return tuple(job)
```

```
# score calculate for experiment not considering PIP score
def _scale_score(score, step):
    __scales = tuple(range(0, step))
    __scores = tuple(score * __scales[i] for i in
range(len(__scales)))
    return __scores
# score calculate for experiment considering PIP score
def _pip_scale_score(score, pip, step):
    __scales = tuple(range(0, step))
    __scores = tuple(score * __scales[i] for i in
range(len(__scales)))
    __pip_scores = tuple(tuple(pip[i] * __scores[j] for j in
range(len(__scores))) for i in range(len(pip)))
   return __pip_scores

# class for experiment not considering PIP score
class Molecule(object):
    def __init__(self, name, vdw, hbd, hba, step):
        self.__name = name
        self.__vdw = _scale_score(score=vdw, step=step)
        self.__hbd = _scale_score(score=hbd, step=step)
        self.__hba = _scale_score(score=hba, step=step)

    def get_name(self):
        return self.__name

    def get_free_energy(self, vdw_weight, hbd_weight, hba_weight):
        __free_energy = self.__vdw[vdw_weight] +
self.__hbd[hbd_weight] + self.__hba[hba_weight]
        return __free_energy

# class for experiment considering PIP score
class Molecule(object):
    def __init__(self, name, vdw, hbd, hba, pip, step):

        self.__name = name
        self.__pip = pip
        self.__vdw = _scale_score(score=vdw, pip=self.__pip,
step=step)
        self.__hbd = _scale_score(score=hbd, pip=self.__pip,
step=step)
        self.__hba = _scale_score(score=hba, pip=self.__pip,
step=step)

    def get_name(self):
        return self.__name

    def get_free_energy(self, vdw_weight, hbd_weight, hba_weight,
pip):
        __free_energy = self.__vdw[pip][vdw_weight] +
self.__hbd[pip][hbd_weight] + self.__hba[pip][hba_weight]
        return __free_energy

# calculate metrics for each result
def cal_metrics(weights, molecules, target):
    output = tuple((molecule.get_name(),
                    molecule.get_free_energy(vdw_weight=weights[0],
hbd_weight=weights[1], hba_weight=weights[2])) for molecule in
molecules)
```

```
    if target == 'SARS-HCoV':
        ranking = return_ranking(output=output, special=True)
    else:
        ranking = return_ranking(output=output, special=False)
    auc_roc = calculate_auc_roc(ranking=ranking)
    auc_bedroc_1 = CalcBEDROC(scores=ranking, col=1, alpha=1)
    auc_bedroc_20 = CalcBEDROC(scores=ranking, col=1, alpha=20)
    auc_bedroc_160 = CalcBEDROC(scores=ranking, col=1, alpha=160.9)
    ef = CalcEnrichment(scores=ranking, col=1, fractions=[0.01,
0.03, 0.0486, 0.06, 0.08, 0.09, 0.12, 0.15, 0.18])
    # if considering PIP
    # result = list(str(weights[3] + 1))
    result = list(cal_ratio(a=weights[0], b=weights[1],
c=weights[2]))
    result.extend([auc_roc, auc_bedroc_1, auc_bedroc_20,
auc_bedroc_160])
    result.extend(ef)
   return tuple(result)

# sub-process calculating results
def multi_cal_metrics(jobs, molecules, target):
    results = []
    for job in jobs:
        result = cal_metrics(weights=job, molecules=molecules,
target=target)
        results.append(result)
    return tuple(results)
def collect_results(results):
    global Results
   Results.extend(results)

if __name__ == '__main__':
    path = 'path of each target"s screened results'
    files = os.listdir(path)
    outputdir = 'path of output file'
    outputfiles = os.listdir(outputdir)
    for file in files:
        Target = file.split('.')[0]
        outputfile = 'name of output file'
        if outputfile in outputfiles:
            print('%s done' % Target)
        else:
            Results = []
            csvfile = path + file
            raw_lines = read_csv(csvfile=csvfile)
            Steps = 21
            Molecules = tuple(Molecule(name=line[0],
vdw=float(line[2]), hbd=float(line[3]), hba=float(line[4]),
step=Steps) for line in raw_lines)
            # if considering PIP
            # Molecules = tuple(Molecule(name=line[0],
vdw=float(line[1]), hbd=float(line[5]), hba=float(line[6]),
pip=tuple(map(float, line[7:])), step=Steps) for line in raw_lines)
            total_weights = create_jobs(step=Steps, reduced=True)
            assignments = split_jobs(jobs=total_weights,
part=mp.cpu_count())
            p = mp.Pool(mp.cpu_count())
            for ass in assignments:
```

```
                p.apply_async(func=multi_cal_metrics, args=(ass,
Molecules, Target), callback=collect_results)
            p.close()
            p.join()
            #not considering PIP
            field_names = ['vdW', 'HBD', 'HBA', 'AUC', 'BEDROC(1)',
'BEDROC(20)', 'BEDROC (160)', 'EF 1%', 'EF 3%', 'EF 4.86%', 'EF 6%',
'EF 8%', 'EF 9%', 'EF 12%', 'EF 15%', 'EF 18%']
            # considering PIP
            # field_names = ['PIP', 'Enthalpy', 'Lost Water
Entropy', 'Side chian Entropy', 'AUC', 'BEDROC(1)', 'BEDROC(20)',
'BEDROC (160)', 'EF 1%', 'EF 3%', 'EF 4.86%', 'EF 6%', 'EF 8%', 'EF
9%', 'EF 12%', 'EF 15%', 'EF 18%']
            with open(outputdir + outputfile, 'wb') as csv_file:
                writer = csv.writer(csv_file)
                writer.writerow(field_names)
                for result in Results:
                    writer.writerow(result)
```

Reference

Ain, Q.U., Aleksandrova, A., Roessler, F.D. and Ballester, P.J., 2015. Machine-learning scoring functions to improve structure-based binding affinity prediction and virtual screening. *Wiley Interdisciplinary Reviews: Computational Molecular Science*. 5(6), pp.405-424.

Amaro, R.E., Baudry, J., Chodera, J., Demir, Ö., McCammon, J.A., Miao, Y. and Smith, J.C. 2018. Ensemble docking in drug discovery. *Biophysical journal*. 114(10), pp.2271-2278.

Ballester, P.J., Finn, P.W. and Richards, W.G. 2009. Ultrafast shape recognition: evaluating a new ligand-based virtual screening technology. *Journal of Molecular Graphics and Modelling*. 27(7), pp.836-845.

Bauer, M.R., Ibrahim, T.M., Vogel, S.M. and Boeckler, F.M. 2013. Evaluation and optimization of virtual screening workflows with DEKOIS 2.0–a public library of challenging docking benchmark sets. *Journal of chemical information and modelling*. 53(6), pp.1447-1462.

Biamonte, M.A., Van de Water, R., Arndt, J.W., Scannevin, R.H., Perret, D. and Lee, W.C. 2009. Heat shock protein 90: inhibitors in clinical trials. *Journal of medicinal chemistry*. 53(1), pp.3-17.

Böhm, H.J., Flohr, A. and Stahl, M. 2004. Scaffold hopping. *Drug discovery today: Technologies*. 1(3), pp.217-224.

Breiten, B., Lockett, M.R., Sherman, W., Fujita, S., Al-Sayah, M., Lange, H., Bowers, C.M., Heroux, A., Krilov, G. and Whitesides, G.M. 2013. Water networks contribute to enthalpy/entropy compensation in protein–ligand binding. *Journal of the American Chemical Society*. 135(41), pp.15579-15584.

Burley, S.K., Berman, H.M., Bhikadiya, C., Bi, C., Chen, L., Di Costanzo, L., Christie, C., Dalenberg, K., Duarte, J.M., Dutta, S. and Feng, Z. 2018. RCSB Protein Data Bank: biological macromolecular structures enabling research

and education in fundamental biology, biomedicine, biotechnology and energy. *Nucleic acids research*. 47(D1), pp.D464-D474.

Campbell, A.J., Lamb, M.L. and Joseph-McCarthy, D. 2014. Ensemble-based docking using biased molecular dynamics. *Journal of chemical information and modelling*. 54(7), pp.2127-2138.

Cereto-Massagué, A., Ojeda, M.J., Valls, C., Mulero, M., Garcia-Vallvé, S. and Pujadas, G. 2015. Molecular fingerprint similarity search in virtual screening. *Methods*. 71, pp.58-63.

Chen, Y.C. 2015. Beware of docking! *Trends in pharmacological sciences*. 36(2), pp.78-95.

Cheng, T., Li, Q., Zhou, Z., Wang, Y. and Bryant, S.H. 2012. Structure-based virtual screening for drug discovery: a problem-centric review. *The AAPS journal*. 14(1), pp.133-141.

De Vivo, M., Masetti, M., Bottegoni, G. and Cavalli, A. 2016. Role of molecular dynamics and related methods in drug discovery. *Journal of medicinal chemistry*. 59(9), pp.4035-4061.

Damm-Ganamet, K.L., Arora, N., Becart, S., Edwards, J.P., Lebsack, A.D., McAllister, H.M., Nelen, M.I., Rao, N.L., Westover, L., Wiener, J.J. and Mirzadegan, T. 2019. Accelerating Lead Identification by High Throughput Virtual Screening: Prospective Case Studies from the Pharmaceutical Industry. *Journal of chemical information and modeling*. 59(5), pp.2046-2062.

Dighe, S.N., Deora, G.S., De la Mora, E., Nachon, F., Chan, S., Parat, M.O., Brazzolotto, X. and Ross, B.P. 2016. Discovery and structure–activity relationships of a highly selective butyrylcholinesterase inhibitor by structure-based virtual screening. *Journal of medicinal chemistry*. 59(16), pp.7683-7689.

Doig, A.J. and Sternberg, M.J. 1995. Side‐chain conformational entropy in protein folding. *Protein Science*. 4(11), pp.2247-2251.

Drwal, M.N. and Griffith, R. 2013. Combination of ligand-and structure-based methods in virtual screening. *Drug Discovery Today: Technologies*. 10(3), pp.e395-e401.

Dunitz, J.D. 1994. The entropic cost of bound water in crystals and biomolecules. *Science*. 264(5159), pp.670-671.

Ehrt, C., Brinkjost, T. and Koch, O. 2016. Impact of binding site comparisons on medicinal chemistry and rational molecular design. *Journal of medicinal chemistry*. 59(9), pp.4121-4151.

Ehrt, C., Brinkjost, T. and Koch, O. 2018. A benchmark driven guide to binding site comparison: An exhaustive evaluation using tailor-made data sets (ProSPECCTs). *PLoS computational biology*. 14(11), p.e1006483.

Empereur-Mot, C., Guillemain, H., Latouche, A., Zagury, J.F., Viallon, V. and Montes, M. 2015. Predictiveness curves in virtual screening. *Journal of cheminformatics*. 7(1), p.52.

Fauchere, J.L. and Pliska, V. 1983. Hydrophobic parameters pi of amino-acid side chains from the partitioning of N-acetyl-amino-acid amides. *European journal of medicinal chemistry*. 18(3), pp.369-375.

Fawcett, T. 2006. An introduction to ROC analysis. *Pattern recognition letters*. 27(8), pp.861-874.

France, D.J., Stepek, G., Houston, D.R., Williams, L., McCormack, G., Walkinshaw, M.D. and Page, A.P. 2015. Identification and activity of inhibitors of the essential nematode-specific metalloprotease DPY-31. *Bioorganic & medicinal chemistry letters*. 25(24), pp.5752-5755.

Ferreira, L., dos Santos, R., Oliva, G. and Andricopulo, A. 2015. Molecular docking and structure-based drug design strategies. *Molecules*. 20(7), pp.13384-13421.

Gaillard, T. 2018. Evaluation of AutoDock and AutoDock Vina on the CASF-2013 benchmark. *Journal of chemical information and modelling*. 58(8), pp.1697-1706.

Gaudreault, F., Chartier, M. and Najmanovich, R. 2012. Side-chain rotamer changes upon ligand binding: common, crucial, correlate with entropy and rearrange hydrogen bonding. *Bioinformatics*. 28(18), pp.i423-i430.

Genheden, S. and Ryde, U. 2015. The MM/PBSA and MM/GBSA methods to estimate ligand-binding affinities. *Expert opinion on drug discovery*. 10(5), pp.449-461.

Halgren, T.A., Murphy, R.B., Friesner, R.A., Beard, H.S., Frye, L.L., Pollard, W.T. and Banks, J.L., 2004. Glide: a new approach for rapid, accurate docking and scoring. 2. Enrichment factors in database screening. *Journal of medicinal chemistry*. 47(7), pp.1750-1759.

Hazuda, D.J., Anthony, N.J., Gomez, R.P., Jolly, S.M., Wai, J.S., Zhuang, L., Fisher, T.E., Embrey, M., Guare, J.P., Egbertson, M.S. and Vacca, J.P. 2004. A naphthyridine carboxamide provides evidence for discordant resistance between mechanistically identical inhibitors of HIV-1 integrase. *Proceedings of the National Academy of Sciences*. 101(31), pp.11233-11238.

Healy, A.R., Houston, D.R., Remnant, L., Huart, A.S., Brychtova, V., Maslon, M.M., Meers, O., Muller, P., Krejci, A., Blackburn, E.A. and Vojtesek, B. 2015. Discovery of a novel ligand that modulates the protein–protein interactions of the AAA+ superfamily oncoprotein reptin. *Chemical science*. 6(5), pp.3109-3116.

Houston, D.R., Yen, L.H., Pettit, S. and Walkinshaw, M.D. 2015. Structure-and ligand-based virtual screening identifies new scaffolds for inhibitors of the oncoprotein MDM2. *PLoS One*. 10(4), p.e0121424.

Ivetac, A. and Andrew McCammon, J. 2011. Molecular recognition in the case of flexible targets. *Current pharmaceutical design*. 17(17), pp.1663-1671.

Jain, A.N. 2008. Bias, reporting, and sharing: computational evaluations of docking methods. *Journal of computer-aided molecular design*. 22(3-4), pp.201-212.

Kairys, V., Fernandes, M.X. and Gilson, M.K. 2006. Screening drug-like compounds by docking to homology models: a systematic study. *Journal of chemical information and modelling*. 46(1), pp.365-379.

Kellenberger, E., Foata, N. and Rognan, D. 2008. Ranking targets in structure-based virtual screening of three-dimensional protein libraries: methods and problems. *Journal of chemical information and modeling*. 48(5), pp.1014-1025.

Kirchmair, J., Markt, P., Distinto, S., Wolber, G. and Langer, T. 2008. Evaluation of the performance of 3D virtual screening protocols: RMSD comparisons, enrichment assessments, and decoy selection—What can we learn from earlier mistakes? *Journal of computer-aided molecular design*. 22(3-4), pp.213-228.

Kolb, P., Huang, D., Dey, F. and Caflisch, A. 2008. Discovery of kinase inhibitors by high-throughput docking and scoring based on a transferable linear interaction energy model. *Journal of medicinal chemistry*. 51(5), pp.1179-1188.

Kollman, P.A., Massova, I., Reyes, C., Kuhn, B., Huo, S., Chong, L., Lee, M., Lee, T., Duan, Y., Wang, W. and Donini, O. 2000. Calculating structures and free energies of complex molecules: combining molecular mechanics and continuum models. *Accounts of chemical research*. 33(12), pp.889-897.

Lagarde, N., Zagury, J.F. and Montes, M. 2015. Benchmarking data sets for the evaluation of virtual ligand screening methods: review and perspectives. *Journal of chemical information and modelling*. 55(7), pp.1297-1307.

Lavecchia, A. and Di Giovanni, C. 2013. Virtual screening strategies in drug discovery: a critical review. *Current medicinal chemistry*. 20(23), pp.2839-2860.

Lim, S.V., Rahman, M.B.A. and Tejo, B.A. 2011. Structure-based and ligand-based virtual screening of novel methyltransferase inhibitors of the dengue virus. *BMC bioinformatics.* 12(13), pp. S24.

Lindh, M., Svensson, F., Schaal, W., Zhang, J., Sköld, C., Brandt, P. and Karlén, A. 2015. Toward a benchmarking data set able to evaluate ligand-and structure-based virtual screening using public HTS data. *Journal of chemical information and modelling*. 55(2), pp.343-353.

Lionta, E., Spyrou, G., K Vassilatis, D. and Cournia, Z. 2014. Structure-based virtual screening for drug discovery: principles, applications and recent advances. *Current topics in medicinal chemistry*. 14(16), pp.1923-1938.

Liu, T., Lin, Y., Wen, X., Jorissen, R.N. and Gilson, M.K. 2006. BindingDB: a web-accessible database of experimentally determined protein–ligand binding affinities. *Nucleic acids research*. 35(suppl_1), pp.D198-D201.

Liu, J. and Wang, R. 2015. Classification of current scoring functions. *Journal of chemical information and modelling*. 55(3), pp.475-482.

Lounnas, V., Ritschel, T., Kelder, J., McGuire, R., Bywater, R.P. and Foloppe, N. 2013. Current progress in structure-based rational drug design marks a new mindset in drug discovery. *Computational and structural biotechnology journal*. 5(6), p.e201302011.

Lyu, J., Wang, S., Balius, T.E., Singh, I., Levit, A., Moroz, Y.S., O'Meara, M.J., Che, T., Algaa, E., Tolmachova, K. and Tolmachev, A.A. 2019. Ultra-large library docking for discovering new chemotypes. *Nature*. 566(7743), p.224.

Martis, E.A., Radhakrishnan, R. and Badve, R.R. 2011. High-throughput screening: the hits and leads of drug discovery-an overview. *Journal of Applied Pharmaceutical Science*. 1(1), pp.2-10.

Matter, H. and Sotriffer, C. 2001. Applications and Success Stories in Virtual Screening. In: Sotriffer. ed. *Virtual Screening: Principles, Challenges, and Practical Guidelines*. Weinheim: Wiley-VCH, pp.319-358.

McClish, D.K. 1989. Analyzing a portion of the ROC curve. *Medical Decision Making*. 9(3), pp.190-195.

Microsoft Corporation. 2016. *Microsoft Excel* (2016). [Software]. [Accessed 20 July 2019]

Morris, G.M., Huey, R., Lindstrom, W., Sanner, M.F., Belew, R.K., Goodsell, D.S. and Olson, A.J. 2009. AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility. *Journal of computational chemistry*. 30(16), pp.2785-2791.

Neudert, G. and Klebe, G. 2011. DSX: a knowledge-based scoring function for the assessment of protein–ligand complexes. *Journal of chemical information and modelling*. 51(10), pp.2731-2745.

Nicholls, A. 2008. What do we know and when do we know it? *Journal of computer-aided molecular design*. 22(3-4), pp.239-255.

O'Boyle, N.M., Banck, M., James, C.A., Morley, C., Vandermeersch, T. and Hutchison, G.R. 2011. Open Babel: An open chemical toolbox. *Journal of cheminformatics*. 3(1), p.33.

Open-source cheminformatics. 2019. *RDKit* (2019.03.1). [Software]. [Accessed 15 June 2019]

Pearson, R.J., Blake, D.G., Mezna, M., Fischer, P.M., Westwood, N.J. and McInnes, C. 2018. The Meisenheimer complex as a paradigm in drug discovery: reversible covalent inhibition through C67 of the ATP binding site of PLK1. *Cell chemical biology*. 25(9), pp.1107-1116.

Plotly Technologies Inc. 2015. *Plotly* (4.1.0). [Software]. [Accessed 15 July 2019]

Ripphausen, P., Nisius, B., Peltason, L. and Bajorath, J. 2010. Quo vadis, virtual screening? A comprehensive survey of prospective applications. *Journal of medicinal chemistry*. 53(24), pp.8461-8467.

Rognan, D. 2013. Proteome-scale docking: myth and reality. *Drug Discovery Today: Technologies*. 10(3), pp.e403-e409.

Sarabipour, S., Ballmer-Hofer, K. and Hristova, K. 2016. VEGFR-2 conformational switch in response to ligand binding*. Elife*. 5, p.e13876.

Schneider, G. 2010. Virtual screening: an endless staircase? *Nature Reviews Drug Discovery*. 9(4), p.273.

Schrödinger, LLC. 2019. *The PyMOL Molecular Graphics System* (2.3.2). [Software]. [Accessed 1 May 2019]

Scior, T., Bender, A., Tresadern, G., Medina-Franco, J.L., Martínez-Mayorga, K., Langer, T., Cuanalo-Contreras, K. and Agrafiotis, D.K. 2012. Recognizing pitfalls in virtual screening: a critical review. *Journal of chemical information and modeling*. 52(4), pp.867-881.

Shave, S., Blackburn, E.A., Adie, J., Houston, D.R., Auer, M., Webster, S.P., Taylor, P. and Walkinshaw, M.D. 2015. UFSRAT: ultra-fast shape recognition with atom types–the discovery of novel bioactive small molecular scaffolds for FKBP12 and 11βHSD1. *PloS one*. 10(2), p.e0116570.

Sheridan, R.P., Singh, S.B., Fluder, E.M. and Kearsley, S.K. 2001. Protocols for bridging the peptide to nonpeptide gap in topological similarity searches. *Journal of chemical information and computer sciences*. 41(5), pp.1395-1406.

Sinha S and Vohora D. 2018. Chapter 2 - Drug Discovery and Development: An Overview. In: Vohora D. and Singh G. eds. *Pharmaceutical Medicine and Translational Clinical Research*. Cambridge: Academic Press, pp.19-32.

Siragusa, L., Luciani, R., Borsari, C., Ferrari, S., Costi, M.P., Cruciani, G. and Spyrakis, F. 2016. Comparing drug images and repurposing drugs with BioGPS and FLAPdock: The thymidylate synthase case. *ChemMedChem*. 11(15), pp.1653-1666.

Śledź, P. and Caflisch, A. 2018. Protein structure-based drug design: from docking to molecular dynamics. *Current opinion in structural biology*. 48, pp.93-102.

Song, C.M., Lim, S.J. and Tong, J.C. 2009. Recent advances in computer-aided drug design. *Briefings in bioinformatics*. 10(5), pp.579-591.

Srinivasan, J., Cheatham, T.E., Cieplak, P., Kollman, P.A. and Case, D.A. 1998. Continuum solvent studies of the stability of DNA, RNA, and phosphoramidate− DNA helices. *Journal of the American Chemical Society*. 120(37), pp.9401-9409.

Sun, H., Pan, P., Tian, S., Xu, L., Kong, X., Li, Y., Li, D. and Hou, T. 2016. Constructing and validating high-performance miec-svm models in virtual screening for kinases: a better way for actives discovery. *Scientific reports*. 6, p.24817.

Taylor, P., Blackburn, E., Sheng, Y.G., Harding, S., Hsin, K.Y., Kan, D., Shave, S. and Walkinshaw, M.D. 2008. Ligand discovery and virtual screening using the program LIDAEUS. *British journal of pharmacology*. 153(S1), pp.S55-S67.

Tian, S., Sun, H., Li, Y., Pan, P., Li, D. and Hou, T. 2013. Development and evaluation of an integrated virtual screening strategy by combining molecular docking and pharmacophore searching based on multiple protein structures. *Journal of chemical information and modelling*. 53(10), pp.2743-2756.

Triballeau, N., Acher, F., Brabet, I., Pin, J.P. and Bertrand, H.O. 2005. Virtual screening workflow development guided by the “receiver operating characteristic” curve approach. Application to high-throughput docking on metabotropic glutamate receptor subtype 4. *Journal of medicinal chemistry*. 48(7), pp.2534-2547.

Trott, O. and Olson, A.J. 2010. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*. 31(2), pp.455-461.

Truchon, J.F. and Bayly, C.I., 2007. Evaluating virtual screening methods: good and bad metrics for the “early recognition” problem. *Journal of chemical information and modeling*. 47(2), pp.488-508.

Vogel, S.M., Bauer, M.R. and Boeckler, F.M. 2011. DEKOIS: Demanding Evaluation Kits for Objective in Silico Screening A Versatile Tool for Benchmarking Docking Programs and Scoring Functions. *Journal of chemical information and modelling*. 51(10), pp.2650-2665.

Walkinshaw, M.D. and Floersheim, P. 1990. Hydrophilicity of proteins and DNA. *Journal of Molecular Structure*. 237, pp.63-73.

Walter, S.D. 2005. The partial area under the summary ROC curve. *Statistics in medicine*. 24(13), pp.2025-2040.

Wang, R., Lai, L. and Wang, S. 2002. Further development and validation of empirical scoring functions for structure-based binding affinity prediction. *Journal of computer-aided molecular design*. 16(1), pp.11-26.

Wang, E., Sun, H., Wang, J., Wang, Z., Liu, H., Zhang, J.Z. and Hou, T. 2019. End-Point Binding Free Energy Calculation with MM/PBSA and MM/GBSA: Strategies and Applications in Drug Design. *Chemical reviews*. No pagination.

Waszkowycz, B., Clark, D.E. and Gancia, E. 2011. Outstanding challenges in protein–ligand docking and structure‐based virtual screening. *Wiley Interdisciplinary Reviews: Computational Molecular Science*. 1(2), pp.229-259.

Wu, S.Y., McNae, I., Kontopidis, G., McClue, S.J., McInnes, C., Stewart, K.J., Wang, S., Zheleva, D.I., Marriage, H., Lane, D.P. and Taylor, P. 2003. Discovery of a novel family of CDK inhibitors with the program LIDAEUS: structural basis for ligand-induced disordering of the activation loop. *Structure*. 11(4), pp.399-410.

Xia, J., Hsieh, J.H., Hu, H., Wu, S. and Wang, X.S. 2017. The development of target-specific pose filter ensembles to boost ligand enrichment for structure-

based virtual screening. *Journal of chemical information and modelling*. 57(6), pp.1414-1425.

Xia, J., Tilahun, E.L., Reid, T.E., Zhang, L. and Wang, X.S. 2015. Benchmarking methods and data sets for ligand enrichment assessment in virtual screening. *Methods*. 71, pp.146-157.

Yang, Y., Moir, E., Kontopidis, G., Taylor, P., Wear, M.A., Malone, K., Dunsmore, C.J., Page, A.P., Turner, N.J. and Walkinshaw, M.D. 2007. Structure-based discovery of a family of synthetic cyclophilin inhibitors showing a cyclosporin-A phenotype in Caenorhabditis elegans. *Biochemical and biophysical research communications*. 363(4), pp.1013-1019.

Zhao, W., Hevener, K.E., White, S.W., Lee, R.E. and Boyett, J.M. 2009. A statistical framework to evaluate virtual screening. *BMC bioinformatics*. 10(1), p.225.

Zheng, H., Handing, K.B., Zimmerman, M.D., Shabalin, I.G., Almo, S.C. and Minor, W. 2015. X-ray crystallography over the past decade for novel drug discovery–where are we heading next? *Expert opinion on drug discovery*. 10(9), pp.975-989.