Deep Neural Networks for Pose Validation, Affinity Prediction, and Input Attribution

Jack Scantlebury

Hertford College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2023

Acknowledgements

It is said that it takes a village to raise a child, and so it goes that it takes a research group to raise a DPhil candidate. First and foremost, I would like to acknowledge my supervisors, Professors Frank von Delft and Charlotte M. Deane, for four-and-a-half years of support and direction. Charlotte has been the sounding board for all of my ideas, and has contributed many of her own; her patience for endless drafting and redrafting has been remarkable. Frank has been an excellent tether between the often remote-seeming worlds of the computer scientist and the biologist, and has helped by ensuring my work is relevant to wider drug discovery community.

Charlotte has cultivated an excellent culture in the Oxford Protein Informatics Group, which has always been a friendly environment, supportive of me and my research. In particular, I would like to thank Tom Hadfield as both a friend and a particularly sharp source of advice and ideas. Ferguses Boyles and Imrie both helped me in the first year, when the learning curve was steepest. For that I am grateful, as I am for Lucy Vost, who co-authored my second publication, taking over the reigns when I disappeared for an internship. Her contributions made for a much stronger paper.

I have made some friends outside of Charlotte's group who deserve an honourable mention. These include Maria Kiourlappou and Patrick Green, my first friends in Oxford, who have been my reliable confidants throughout. My best friends from Hertford College, Niall Maher and Hans Erik Havsteen, made my final year my favourite.

Finally, I would like to thank Anna, for her unwavering support, as well as my parents and my siblings. You have each played your part in various ways, and you are all very dear to me.

Abstract

The search for drug molecules which bind strongly to specific proteins is an integral part of the drug discovery process. To this end, virtual screening algorithms which aim to screen a large number of potential binders *in silico* have been developed. These use scoring functions to assess the probability that a computationally predicted binding pose is correct, and to predict the binding affinity. More recently, research has turned to deep learning-based scoring functions which use binding data to build a model of binding behaviour; this is the main subject of this thesis.

The first chapter is an introduction to the concepts and literature which are relevant to the subsequent chapters. This includes fragment-based drug discovery, virtual screening, and machine learning methods in virtual screening. It also touches on the problem of input attribution, where importance is assigned to the atoms or bonds in the input to deep learning-based scoring functions, as well as the problem of machine learning algorithms learning to classify based on dataset biases rather than learning the physical interactions which govern protein-ligand binding.

A publication on convolutional neural networks for virtual screening makes up most of the second chapter. The problem of learning training set biases rather than physical interactions is explored using several experiments, and a method of dataset augmentation to combat this effect is proposed. A carefully curated validation set, constructed separately from any training data, is used to show increased use of protein information in classification decisions; input attribution is used on several case studies to show the same.

The third chapter concerns the design and engineering decisions involved in PointVS. This is a software package for rapid prototyping and testing

of graph neural networks for pose classification and affinity prediction. It includes a variety of scripts for auxilliary tasks such as dataset generation, input attribution visualisation and logging, and has been used by another member of the Oxford Protein Informatics Group for a paper which is briefly described.

PointVS forms the basis of another publication, under review at the time of writing; this makes up the fourth chapter. In collaboration with another student, graph neural networks are used for pose classification and affinity prediction, with training and testing sets set up carefully to avoid information leakage. PointVS is compared to several other methods, achieving competitive performance. Attribution scores from PointVS are converted to protein hotspot maps, which are used as input into a generative model for fragment elaboration. This out-performs the results using standard physics-derived hotspot maps, which is evidence that the graph neural networks can pick out important protein-ligand interactions.

Finally, we take a more macroscopic view of the field of machine learning-based scoring functions. We conclude that while promising, there are several obstacles that must be overcome in order to train models which truly 'understand' the universe of protein-ligand binding. We propose that an input attribution ground truth test set as an area for possible further study, and identify a possible method to generate one. We conclude that many reported improvements of machine learning-based scoring functions over their physics-inspired predecessors are over-estimated, and that a more dynamic view of binding which takes water into account explicitly is required.

Contents

List of Figures

Abbreviations

Notation	Description	Page List
CASF	Comparative Assessment of Scoring Functions	8, 10–12, 96
CNN	convolutional neural network	ii, v, vii, viii, 10–12, 14–17, 24, 27, 29–33, 35, 38–51, 63, 67, 72, 74, 82, 96
CSAR	Community Structure Activity Resource	9, 10
DFT	density-functional theorem	16, 95
DUD-E	Directory of Useful Decoys - Extended	vi, vii, 6, 10–12, 14, 24, 27, 29, 30, 32–35, 37–39, 41–43, 45–47, 51, 62, 97
EGNN	$E(n)$-Equivariant Graph Neural Network	vi, ix, 21, 22, 25, 54, 63, 68, 70, 75, 92
FBDD	fragment-based drug discovery	2, 3, 23, 67
GNN	graph neural network	15, 16, 20, 24–26, 51, 54, 61, 62, 67, 72, 82, 94, 96, 117
GPU	graphics processing unit	9
LBVS	ligand-based virtual screening	4, 9
ML	machine learning	8, 9, 12–14, 53, 61, 62, 64, 66, 96–98
PDB	Protein Data Bank	viii, 12, 14, 40, 49, 57, 59, 72, 73, 85, 86, 92
PR-AUC	area under the of the precision-recall curve	41, 43, 51
ROC-AUC	area under the curve of the receiver-operator characteristic	10, 41, 51, 96
SBVS	structure-based virtual screening	4, 9–11, 13, 14, 24, 26, 28, 30, 94, 96
SVM	support vector machine	9

Notation	Description	Page List
VS	virtual screening	3, 4, 10, 24, 28, 29, 31

Chapter 1

Introduction

The process of tackling symptoms of a disease can be summarised, in the broadest terms, as follows: a collection of symptoms with a common cause are recognised, after which the biological pathway which causes the symptoms are identified. A protein which forms part of the pathway is designated the target, for which a drug should be found. At this stage, the drug discovery process can begin. This laborious and expensive part of the puzzle can be split into the 'early', 'pre-clinical' and 'clinical' stages. Early drug discovery is the concern of this thesis; this is where a small number of lead compounds are identified for testing in pre-clinical and clinical trials on animal and human subjects.

A molecule must be found which will modify, *in vivo*, the biological pathway, treating or curing the malady. In this context, modification usually means inhibition by binding strongly to the orthosteric or an allosteric site of a protein. The molecule should not interrupt other pathways, but it should be synthetically feasible, non-toxic, and biologically available. In order to achieve this we must identify the appropriate molecule from the vast chemical space (possibly as high as 10^{60}) of drug-like compounds [1]. The median (mean) cost of this has been put at $985 million ($1,336 million) [2].

We can conceive of two measures for the efficiency of the early drug discovery process: the cost (in time and labour) of discovering a compound for use in trials, and the probability of that compound fulfilling all requirements for clinical approval. Often these are competing considerations; choosing a compound at random from a library has no cost, but almost zero chance of success and so the expensive trial stage is wasted. Putting all synthetically accessible drug-like compounds through assays and cell cultures to measure affinity would likely produce a promising lead, but clearly that is not feasible.

Two major innovations have begun to revolutionise the drug discovery process, and their combination is a natural avenue of research. These are fragment-based drug discovery and computational methods in drug discovery.

1.1 Fragment-Based Drug Discovery

The infeasibility of enumerating every drug-like molecule has led medicinal chemists to invent a way of modularising the search into simpler subtasks, the results of which can then be combined, analogous to recursive programming in computer science. In 1996, Shucker *et al.* discovered multiple nanomolar binders for the FK506 binding protein by merging fragments with micromolar affinities, using what they termed 'SAR by NMR' [3], in a work widely seen as the birth of a technique now known as fragment-based drug discovery (FBDD). The process is outlined in Fig. 1.1.

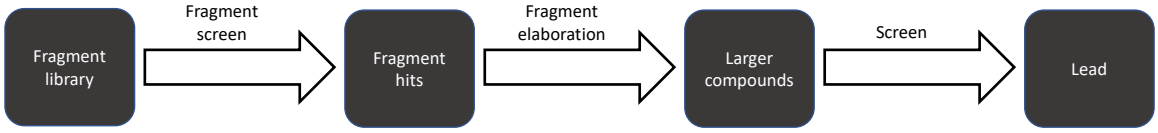


Figure 1.1: Workflow for fragment-based drug discovery. A library of fragments (usually on the order of hundreds of molecules) is screened against the target in question. The binding modes of the fragments on the site of interest are ascertained, usually using X-ray crystallography. These hits are used to design larger molecules, which conserve the binding interactions of the fragment hits. Depending on the number of elaborations, the resulting larger molecules can be virtually screened before being screened in a lab, where binding affinity can also be measured.

In FBDD, target proteins are soaked in smaller molecules ('fragments'), binding modes and affinities are determined. Multiple fragment 'hits' can be combined, or single hits can be grown, until a drug-sized molecule ('lead') is arrived at. In this way, the search space is vastly smaller, and much more easily covered by a fixed library of perhaps 100-1000 fragments. There are three main ways in which fragment hits are used to find leads [4]:

1. Growing: a single fragment is extended from one or more synthetically accessible sites, usually with the goal of adding a compatible functional group to a nearby protein pharmacophore.

2. Linking: two or more fragments are linked together by one or more bridging atoms.

3. Merging: two or more fragments which showed some overlap in binding mode are merged together in a synthetically sensible way.

In order to make informed choices, each of these methods requires high resolution structures, as well as accurate affinity data. Usually binding affinities are much lower than those for drug-like molecules and leads, and the molecules can bind at multiple sites, so accurately measuring the strength of interaction is difficult. Methods which work well for lead-sized molecules are sometimes not sensitive enough to measure the strength of protein-fragment interactions [5], such as Isothermal Titration Calorimetry.[1] Ideally, two or more orthogonal methods are used to confirm affinity, such as ^{19}F-NMR [7] and Differential Scanning Fluorimetry [8].

Binding modes can be ascertained in several different ways; while X-ray crystallography remains popular, recent advances include high-throughput broadband ^{19}F-NMR [9], weak-affinity chromatography [10], and cryo-EM [11]. All structures used or discussed in this thesis sets have been obtained using X-ray crystallography.

Both binding mode and affinity discovery can be aided by computational methods. The static picture of binding given by very low temperature crystals is not always reflective of the rich energy landscape which describes the behaviour of protein-fragment and protein-ligand complexes. Modifications to known binders can be tested, with varying levels of accuracy, *in silico* before expensive lab-based experiments and there exists an ever-growing suite of algorithms to inform, or even replace, the human element of the fragment-to-lead process.

1.2 Computational Methods in Drug Discovery

While not currently as accurate as lab science [12], computational drug discovery is orders of magnitude faster and can be used to narrow down the number of compounds tested *in vitro* to a manageable number, in a process known as virtual screening (VS). Recalling the two measures for efficient early drug discovery, computational methods which are faster will take less time to suggest leads, and those which are more 'accurate' are more likely to include in their budget of suggested compounds one which either fulfil or can be modified to fulfil the biophysical requirements.

[1] While commercially available ITC methods are not usually used in FBDD due to their throughput and sensitivity limitations, there is mounting evidence that the binding strength of the best drugs is chiefly enthalpic rather than entropic [6], something ITC is particularly suited to detecting.

In 2019, Lyu *et al.* computationally screened 138 million drug-like chemicals against the D_4 dopamine receptor, finding an 180 pm agonist alongside 30 other sub-micro molar binders [13]. This took over 43,000 core-hours but with 150 cores (a not unreasonable number for any well resourced research group or pharmaceutical company), this is reduced to under 13 days. Screening this many compounds in a wet lab is out of the question, illustrating the growing appeal of virtual screening in drug discovery. For examples of recent applications of computational drug discovery, reviews can be found at [14, 15].

There is a vast array of free and proprietary software available for VS [16]. These can be categorised in several different ways. One of the more useful distinctions is that between structure-based virtual screening (SBVS) and ligand-based virtual screening (LBVS). SBVS requires a 3D structure of the target protein, ideally with atomic-level precision. It attempts to find a strong binder by modelling the physical interactions between the protein and its ligand, often by simulating a binding event. On the other hand, LBVS only requires a known (possibly weak) binder, used as a 'jumping off' point from which other, related ligands can be engineered to bind more tightly. LBVS is inspired by the observation that similar ligands show similar binding activity for the same protein [17].

The merits of both approaches are scenario specific. LBVS is generally faster than SBVS, so more potential ligands can be screened, but it is limited because the topology and energetics of the binding site are vital to binding behaviour [18]. If there are no known binders, LBVS cannot be used as there is no starting point; if the structure of the target protein has not been solved, SBVS cannot be used. The corollary is the combination of the two methods, where initial starting points for LBVS are generated by SBVS, a procedure which can be iterated on. A review of this hybrid approach can be found at [19]. This thesis will be concerned only with structure-based methods; for a review of ligand-based methods, see [20].

There is a rich array of possible representations of the 3D structural representation in SBVS, especially with regards to the protein. This can range from simply reducing amino acids to their amino acid sequence [21], to treating each atom as a particle in a dynamic force-field [22], approximations of the full binding site electron density [23, 24], and even quantum mechanical representations [25].

The two most commonly used methods in SBVS are docking and molecular dynamics [26]. Here I will present a short review of the state of these methods, their advantages, and their shortcomings as applied to SBVS.

1.2.1 Subtasks of Structure-Based Virtual Screening

SBVS consists of three related tasks:

1. The search algorithm samples the energy landscape to find minima with respect to the binding energy.

2. The scoring function ranks the resultant poses to choose the 'correct' one for each ligand.

3. The top-ranked poses for each ligand are ordered by binding affinity so the strongest binding ligands can be selected.

The first of these tasks is delegated to AutoDock Vina [27] throughout this thesis, but is vital to any virtual screening package. Tasks 2 and 3 (the subject of this thesis) are impossible if a pose which is close to what would be observed experimentally is not included in the output of the search algorithm.

1.2.2 Docking

Docking is a two-stage algorithm: poses are first searched for and then ranked. Its target is the pose a ligand would take when bound to the target protein. It also usually estimates the binding energy of the interaction, taking into account interaction energies using an approximation to the binding affinity called a scoring function. Scoring functions come in four main categories: physics-based, knowledge-based, empirical, and machine learning-based. It should be noted that the minimisation can be with respect to another function, where the scoring function is used only to score and rank the final poses as in Quantum Mechanical Docking Scoring [28]. The search algorithm can also be a direct minimisation of the scoring function as in AutoDock Vina [27] where during the search phase, ligand geometry is subject to random mutations and local minimisations which are accepted with a probability proportional to the associated change in score. For a more thorough explanation of the search algorithm and scoring function used by AutoDock Vina, see Section 1.2.2.4.

1.2.2.1 Physics-based Methods

Physics-based methods can be split further into force field-based and quantum mechanical. In the former, the binding affinity ΔE_{bind} is calculated by the addition of energy terms from a classically-derived force field. This usually includes intermolecular interactions such as electrostatic ($\Delta E_{electrostatic}$), van der Waals (ΔE_{vdW}),

hydrogen bonds (ΔE_{H-bond}), steric, as well as intra-molecular interactions (ΔE_{intra}) such as torsional strain and internal sterics. Some modern methods also include a term for solvation energy:

$$\Delta E_{bind} = \Delta E_{solvation} + \Delta E_{vdW} + \Delta E_{electrostatic} + \Delta E_{H-bond} + \Delta E_{intra}$$

These were among the first scoring functions developed [29, 30], and are often fitted to experimental data (see AutoDock 3 and AutoDock 4 for prominent early examples [31, 32]). Unfortunately, accounting for every interaction is difficult. Quantum mechanical effects are omitted, and other non-linearities are ignored. Popular force field-based scoring functions include CHARMM [30], AMBER [33] and GoldScore [34].

Physics-based quantum mechanical methods aim to offer a higher accuracy than force-field methods, at a much higher computational cost. This can be conducted using Wavefunction methods (such Hartree-Fock and its derivatives), but density functional theory [35] is usually used due to its relative speed when dealing with large systems. The cost is still substantially greater than for force field-based methods, so running quantum mechanical calculations on millions of protein-ligand systems, each with multiple possible binding modes, is not currently feasible. Quantum mechanical scoring functions have been used to separate small numbers of actives and decoys from the DUD-E [36] dataset on a handful of proteins [28], exhibiting greater discriminative power than a standard method such as AutoDock Vina.

1.2.2.2 Knowledge-based Methods

Knowledge-based methods build a profile of intermolecular interactions from solved structures with known binding affinities. Pairwise interactions x_{ij} are calculated as:

$$x_{ij} = -k_B T \log \frac{\rho_{ij}(r)}{\rho_{ij}*}$$

where k_B is the Boltzmann constant and T is the temperature. The strength of an interaction is taken as the difference between the smoothed density of that particular combination of atoms at distance r, $p_{ij}(r)$ and some reference density where the pair of atoms are assumed not to be interacting $p_{ij}*$, a concept taken from statistical mechanics in liquids. The resulting statistical potential is then applied to unknown protein-ligand combinations. The main difference between different knowledge-based scoring functions is in how the pairwise and reference densities are calculated.

It has been pointed out that a Boltzmann distribution for all ligand-atom combinations cannot be assumed [37]. Pairs of atoms also affect the presence of other

atoms and combinations, for example in functional groups, which is not accounted for. Knowledge-based scoring functions are, however, computationally inexpensive once the densities have been derived from some corpus of previously solved structures. Difficult-to-calculate terms such as solvation are captured implicitly; knowledge-based scoring functions can be seen as the simpler, statistical cousin of machine learning-based scoring functions. Examples of knowledge-based scoring functions include DrugScore [38], PMF [39] and IPMF [40].

1.2.2.3 Empirical Methods

Empirical methods have their roots in a 1994 work by Bohm [41], a scoring function still in use as part of BIOVIA Discovery Studio [42]. This scoring function is a linear combination of contributing terms c_i:

$$\text{Score}_{emp} = g\left(\sum_i \phi_i c_i\right)$$

where ϕ_i are scalar weights to be learned, and g is usually the identity function. Various properties of the interaction c_i are calculated for the protein-ligand complex. In the simplest case, this could be 1D descriptors of the interaction such as the numbers of hydrogen bonds, fixed rotable bonds, non-polar interactions, etc., or more complex pair- or even n-wise properties of individual atomic interactions. Regression on some set of solved structures with known binding affinities is used to find linear weights ϕ_i for each term. Accuracy on new systems is therefore dependent on how similar these new systems are to the data to which the weights were fitted.

One of the most widely-used empirical scoring functions is that found in AutoDock Vina [27], itself derived from X-Score [43]. AutoDock Vina is the docking algorithm used to generate all poses for the work in this thesis, so it is worth delving into the specifics of its scoring function, its search algorithm, and its strengths and weaknesses.

1.2.2.4 AutoDock Vina

The scoring function for AutoDock Vina is relatively simple. Each pair of atoms is assigned a value for five interaction terms as a function of the distance between them: three based on sterics, one for hydrogen bonding, and one for hydrophobic interaction strength, indexed by k in the overall score:

$$\text{Score}_{ADV} = \frac{1}{1 + \phi_6 N_{rot}} \sum_{i<j} \sum_{k=1}^{5} \phi_k c_{k,ij}(r_{ij})$$

where $c_{ij,k}(r_{ij})$ is the k^{th} pairwise interaction type between atoms i and j as a function of their separation r_{ij}, and N_{rot} is the number of rotable bonds in the ligand. The weights of each type of interaction $\{\phi_1, ..., \phi_6\}$ are obtained by regression on the PDBBind refined set [44], which is the part of the PDBBind containing only high-quality protein-ligand structures and their affinities. While both the inter- and intra-molecular parts of the scoring function in AutoDock Vina are minimised during the search phase, the final pose ranking only depends on intermolecular terms.

During optimisation, an Iterated Local Search [45] is used to find low energy poses, in which a random mutation is made to the ligand geometry, followed by a local minimisation, repeated until convergence. The Broyden-Fletcher-Goldfarb-Shanno quasi-second order minimisation algorithm [46] approximates the Hessian of the ligand atom positions with respect to the scoring function, so a minimal number of steps are needed to achieve an acceptable local minimum at each step. The mutation is then accepted with a probability according to the Metropolis criterion [47], using the scores of the current and prospective poses as arguments.

The designers of the Comparative Assessment of Scoring Functions (CASF) placed the AutoDock Vina scoring function as among the best overall non-machine learning scoring functions across their four benchmark tests (scoring, ranking, docking and screening power) [48], and the best machine learning scoring function ($\Delta_{\text{Vina}}\text{RF}_{20}$) is a reparameterisation of the AutoDock Vina scoring function [49]. In a study comparing five docking programs, it was found that AutoDock Vina found high affinity binders at a rate similar to both GlideScore, another empirical scoring function with terms for lipophilic enclosure of ligand motifs, water desolvation energy fitted to $\sim$ 200 protein-ligand complexes [50]. AutoDock Vina also outperformed all other scoring functions at finding poses similar to the crystal pose, while being significantly faster [51]. The difference in performance between the two tasks suggests AutoDock Vina is a good target for rescoring functions.

1.2.3 Supervised Machine Learning

If we assume that binding affinity at a specific temperature is a deterministic function $f^*(p, l) \rightarrow \mathbb{R}$ of the protein-ligand complex (p, l) to a real number on $\mathbb{R}$ representing the binding energy, then the aim of supervised machine learning (ML) is to learn an approximation to this function parametrised by $\boldsymbol{\theta}$, $f(p, l; \boldsymbol{\theta})$. Protein-ligand complexes are dynamic systems, constantly cycling through various high- and low-energy states, so the ideal ML function would be able to infer this, along with the corresponding

effect on affinity, from the bound crystal state, having learned the perfect binding model, $f = f*$. *In lieu* of this, we would like to learn $f* \approx f$.

This has been an active pursuit of scientists for several decades. A search for 'protein' and 'machine learning' on the Web of Science database [52] yields just a single result before 1990, 60 results for 1990-1999, 1,315 for 2000-2009, and 6,362 for 2010-2019. Older ML algorithms used hand-engineered features as inputs into, for example, a random forest or support vector machine (SVM). These features could come from the ligand only (analogous to LBVS), or from both the protein and the ligand (SBVS), such as the linear regression-based scoring functions employed by AutoDocks 3 [32], 4 [31] and Vina [27]. Some of the older representations were not designed to account for interactions at the atomic level, and these have largely been superseded by a constellation of increasingly exotic neural network architectures that have come from the computer science community in the last decade. This thesis is not concerned with these 'classical' methods, but I will include a brief background of their historical application to protein-ligand interactions. The more relevant neural networks will then be detailed, as well as some of the more notable examples of their application to drug discovery. For a review of the history of ML in organic chemistry, including biochemistry, see [53].

1.2.3.1 Methods Other Than Deep Neural Networks for Binding Affinity and Pose Prediction

Before the rise of GPU-accelerated deep learning, machine learning methods in drug discovery (and medicine more widely) tended to rely mostly on hand-engineered features as input to classical ML algorithms. The advantage of these methods is that they usually have considerably fewer parameters than deep neural networks, need less training data, and are more interpretable. This comes at the expense of expressiveness and the sensitivity to the choice of hand-crafted input features.

As late as 2011, support vector regression (an extension of the SVM) was used as a scoring function which performed competitively (at the time) on the Community Structure Activity Resource (CSAR) 2010 protein-ligand affinity test set [54] when features were taken from the 3D structure [55]. Eighty-four million potential anti-cancer compounds were screened using a SVM to identify several hundred potential drugs, among which were several already-approved treatments [56]. For a review on SVMs and their application to drug discovery, see [57].

Random forests [58] are a popular choice of classical machine learning algorithm for both classification and regression. They benefit from ensemble learning (a 'bag

of decision trees'), as well as interpretability, but still rely on hand-crafted input features. They have been used to predict the sensitivity of cancer cells to certain drugs [59, 60], to derive a structure-activity relationship for VS [61]. A random forest trained on features extracted from the AutoDock Vina representation of protein-ligand complexes [49] came first in the *scoring power* test of the CASF 2016 update [48].

1.2.3.2 Convolutional Neural Networks

The CNN has been a staple of all forms of computer vision since its first widely-publicised application to image recognition in 1995 [62]. Convolutional layers are inspired by the neuroscientific concept of the receptive field. The values of later layers are only a function of the weights of the region of the previous layer which are in the same spatial neighbourhood, rather than all previous neurons as in a fully connected layer. This has two benefits. Firstly, the number of parameters per layer is drastically reduced, because the weights are shared for each region of the image. Secondly, it introduces translational invariance, in which the prediction does not change when in the input image is translated relative to the input frame. This a component of the $E(3)$ symmetry group to which SBVS algorithms are ideally invariant. What follows is a discussion of a small number of important CNNs for SBVS published in the last five years.

Gnina. CNNs are not only applicable to 2D images. Protein-ligand complexes can be conceived of as 3D continuous images, with input atoms represented by different channels (analogous to the red, green and blue channels in a colour image). This is the approach adopted by the authors of Gnina, who use a pseudo electron density to discretise atomic coordinates into 3D voxel values. In this way, a rich 3D representation of the protein-ligand complex is constructed, and image recognition networks can be used for tasks such as pose classification or affinity regression, as illustrated in Fig. 1.2.

In the original work [24], the task was virtual screening on the DUD-E dataset[2] [36], as well as pose prediction. On the CSAR [54] inter-target pose prediction test, Gnina achieved an area under the curve of the receiver-operator characteristic (ROC-

[2]For a more detailed description of DUD-E, see the Methods/Datasets section of the paper which makes up most of in Chapter 2, Section 2.2.1.

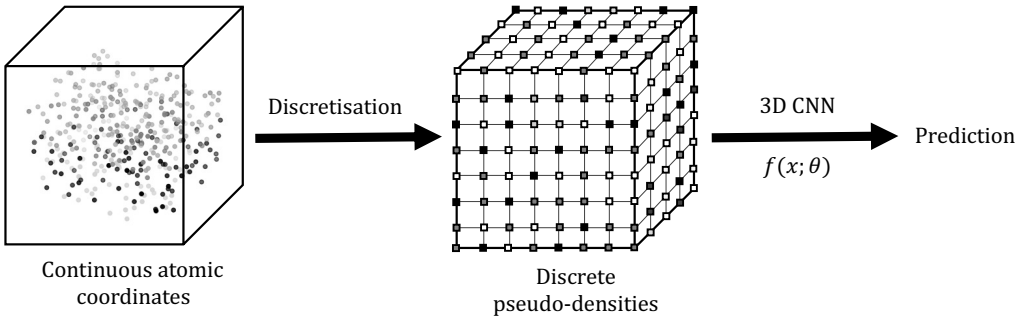


Figure 1.2: Discretisation and prediction with Gnina. (Left to centre) Discretisation, from atomic positions to pseudo-electron densities. The distance from each discrete coordinate to each atom is computed, and a pseudo-electron density is calculated using a piecewise combination of a Gaussian centred on the exact atomic position (within the van der Waals radius of the atom type) and a quadratic (beyond the van der Waals radius). The value in each channel at each discrete voxel is the sum of these densities over all atoms of the type associated with the channel. The central image is an example of values for just a single channel, with different shades indicating different pseudo-densities. (Centre to left) Prediction of pose score or affinity using a CNN (f, parameterised by model weights θ) on the 3D discretised image (x).

AUC) of 0.815, compared to 0.645 for AutoDock Vina, but this is reversed (Top-1[3] of 64% vs 84%) for intra-target pose ranking, which is more relevant to real-world drug discovery campaigns. For virtual screening, where affinity is predicted using the scoring function, Gnina performed better than AutoDock Vina on 90% of DUD-E targets. Although careful attention was paid to clustered cross-validation during training and testing on the DUD-E dataset, there are fundamental limitations on measuring generalisability using this particular dataset, which will be discussed later in Chaper 2 (Section 2.1). The first paper published during the research that went into this thesis was based on the Gnina representation of the protein-ligand complex [63].

Pafnucy. Another influential CNN algorithm for SBVS is Pafnucy [64]. The idea was similar to Gnina: a 3D representation with different features in a 4^{th} channel dimension. The discretisation went the route of moving atoms to their nearest 1 Å voxel, removing the pseudo-density and treating atoms as particles on a grid. When trained and tested on different parts of the PDBBind database v.2016 [65] as part of CASF *scoring power* test, they achieved a Pearson's Correlation Coefficient of 0.70,

[3]Top-N scores are the rate at which a 'good' pose (usually defined as within 2 Å of the crystal pose) are found in the top N poses, as ranked by some scoring function. Top-1 is the rate at which a good pose is ranked best by the scoring function.

which according to the authors was the second-best published CASF score at that time (2018). No attempt was made to prevent information leakage between the test, train and validation sets other than removing identical structures from the training set, something which which will be discussed in more detail in Chapter 4, Section 4.3.1.

DeeplyTough. While not a pose classifier or scoring function itself, DeeplyTough represents an interesting use of CNNs for structure-based drug discovery, and is one of the first deep neural networks in which explicitly accounts for symmetries other than translational [66]. Protein pockets are given vector embeddings by a CNN, trained by rewarding similarity between embeddings of similar pockets and penalising similarity between embeddings of dissimilar pockets. DeeplyTough was trained and tested on splits of ~ 1 million protein pairs that make up TOUGH-M1 [67], a dataset curated from the Protein Data Bank (PDB) [68] where binding site topology as well as the ligands which bind the proteins were used to cluster the pockets, drawing similar and dissimilar pairs from the resultant groupings. TOUGH-M1 was carefully split into a training set and a test set, ensuring that no two proteins with more than 30% sequence similarity were in the opposite split. Once the network is trained, obtaining the similarity between pockets is computationally inexpensive. All-to-all comparisons on a large number of pockets benefit particularly from this, because after the forward pass is executed to find the embedding of each protein once, it can be reused to calculate similarity with other embeddings at very low cost (using the Euclidean norm). 3D steerable CNNs [69] are used to give rotational invariance to the predictions with respect to the input atoms.

DeeplyTough achieved excellent performance when tested on different folds of the training set, mirroring performance of Gnina on DUD-E; however, as with Gnina, generalisability was shown to be a more difficult challenge, with the CNNs achieving results significantly worse than several other non-ML methods when tested on externally constructed validation sets. The authors discussed the problems in constructing datasets for ML-based pocket matching, chiefly among them the rate of false negatives, which they posit is an inherent flaw in the method by which TOUGH-M1 is constructed.

Since the publication of these CNNs, it has also become apparent that care has to be taken to avoid using learning ligand biases to predict properties of protein-ligand complexes.

1.2.3.3 Overfitting, or Learning from Dataset Biases Rather Than Physical Interactions

An ongoing theme in ML for SBVS is the failure of the resulting scoring functions to generalise to external validation sets [70]. Generalisation error is the difference in performance between that found on the training set, and that for the true underlying distribution. Good training data covers the underlying distribution well. In the case of scoring functions, this means that training data should contain a diverse array of protein-ligand complexes, distributed such that the the space of all possible protein-ligand combinations is well-represented. Using this data, a well-engineered model would be able to achieve a small generalisation error. A large generalisation error is indicative of overfitting to the training data, where model bias is low but variance is high.

Dataset Size. In the simplest case, with no constraints on the weights, a polynomial of degree n can be made to perfectly fit $n+1$ points. For example, a straight line defined by $y = \theta_0 x^0 + \theta_1 x^1$ can be made to perfectly coincide with any two points on a plane, simply by altering θ_0 and θ_1. We can keep doing that for more points, using more complex polynomials, and the error in fitting the line to the points can be kept at zero, but we are not learning anything useful about the relationship between x and y. The model is unlikely to accurately predict y for new examples of x. This is the concept of overfitting, and comes about because we do not have enough data to train a complex model.

The same principle applies to all supervised learning algorithms, including neural networks. With increasing model complexity comes a need for more training data to avoid overfitting, and high generalisation error. There are many methods to combat overfitting, usually by placing constraints on the learned weights. This can be as part of the loss function, where regularisation penalises large model weights, by weight decay in which model weights decrease consistently throughout training, or by using dropout, in which a some weights are ignored at random during training, learning in effect an ensemble of models [71].

As applied to ML-based scoring functions, the problem is highlighted by the size of the most common training set for affinity data, the PDBBind [44]. In the most recent release (v.2020), there are 19,443 protein-ligand complexes complete with binding affinity. The size of modern deep neural networks is usually measured in millions of parameters. Great care must be taken to avoid overfitting on PDBBind. Also, the

data is not evenly spread within PDBBind, as research tends to be more concentrated on certain families of proteins of interest to science and medicine, so a relatively small pool of scaffolds dominates publications in organic chemistry [72].

Learning Ligand Bias. The concentration of structures from the PDB around a small number of regions of chemical space gives rise to models which perform poorly on truly novel proteins and ligands. In [73], the authors conclude that the largest (at the time) virtual screening training set, DUD-E, is biased to such an extent that cross-validated performance comparable to SBVS CNNs can be achieved by training a random forest on the 1D ligand properties against which the dataset was explicitly de-biased. This problem is because active ligands (ligands which bind to the protein) can be separated from decoys (ligands which do not bind) without making use of information from the protein, and without learning specific protein-ligand interactions [74].

Learning Protein Bias. DUD-E contains 102 protein targets, of which 25 are kinases and 15 are proteases. It is perhaps unsurprising, then, that CNNs trained on DUD-E perform better on these protein classes than on other proteins [63, 23]. Scientists are more interested in some proteins than others, usually due to their association with disease and their druggability, so research efforts are not evenly distributed across protein space. The resulting imbalance in solved structures causes biases in ML methods for SBVS. It also implicitly causes ligand bias, as ligands which bind to over-represented protein families will appear more in training sets than those which are specific to proteins which are less commonly studied at a structural level.

This is a finding that has since been repeated [63, 75], and is a topic of active research. Chapter 2 of this thesis is based on a publication which seeks to remedy this using dataset augmentation, and Chapter 3 touches on a paper I wrote code for which uses synthetic data to probe the robustness of various ML methods to ligand bias, specifically using the DUD-E and LIT-PCBA [76] datasets as examples.

1.2.3.4 Graph Neural Networks

While protein-ligand complexes can be voxelised into 3D images appropriate for input into a CNN, discrete voxels are not a natural way to represent the coordinates of an atom $\overrightarrow{x} \in \mathbb{R}^3$. The concept of the bond is also entirely absent from CNN representations, and the idea that some atoms are covalently bonded to one another must

be inferred from scratch during training. Representing a molecule as a graph solves both of these problems. Graph neural networks (GNNs) are a family of deep neural networks which operate on graphs, and were a later development than CNNs. GNNs were invented in 2009 [77], but the innovation which motivated the explosion in the use of GNNs for a wide variety of tasks was the graph convolution layer, submitted to arXiv by Kipf and Welling in 2016 [78].

The input to a GNN is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consisting of a set of N vertices $\mathcal{V} = \{v_i\}_{i=1}^{N}$ connected by M edges $\mathcal{E} = \{e\}_{i=1}^{M}$, where each edge in $\mathcal{E}$ is between two vertices in $\mathcal{V}$ and can be represented by a matrix $A \in \mathbb{R}^{N \times N}$, whose values are zero for unconnected nodes and either 1 or some weight for connected nodes. At each successive layer of the graph, 'messages' are passed between adjacent nodes, such that information between nodes propagates outwards at each successive layer. By the l^{th} layer, information from a node has reached nodes at l degrees of separation. Signals from nearby nodes have a stronger effect than signals from distant nodes, analogous to the chemical environment of a ligand atom being affected more by its neighbours than by atoms which are further away. In general, the embedding of the i^{th} node at layer l, $\boldsymbol{h}_{i,l}$, is updated for layer $l+1$ according to:

$$\boldsymbol{h}_{i,l+1} = g\left(\boldsymbol{h}_{i,l}, f\left(i, \{\boldsymbol{h}_{j,l}\}_{j\neq i}^{N}, A\right)\right)$$

where f is an aggregation function, and g is the update function. f must be permutation invariant, so a simple aggregation function can be the sum of neighbours weighted by their edges from A:

$$f\left(i, \{\boldsymbol{h}_{j,l}\}_{j\neq i}^{N}, A_l\right) = \sum_{j\neq i}^{N} A_{ij}\boldsymbol{h}_{j,l}$$

The arguments for f include the set of all other node embeddings $\{\boldsymbol{h}_{j,l}\}_{j\neq i}^{N}$ and a connectivity matrix A, representing the edges, which modulates the rate of propagation of messages from neighbouring nodes. There are multiple ways to represent A as a square matrix (these representations are connected by $L = D - J$):

- Degree matrix (for an undirected graph) D, which is diagonal with D_{ii} holding the number of edges connecting node i. Note that the graph cannot be represented entirely by D, as information about specific edges is lost.
- Adjacency matrix J, where $J_{ij} = 1$ if an edge exists between nodes i and j or zero otherwise.

- Laplacian matrix L, where $L_{ij} = Dij$ if $i = j$, -1 if there is an edge between nodes i and j, or zero otherwise.

The update function g could then also be a simple addition of the embedding and message vector $\boldsymbol{m}_{i,l}$ resulting from f:

$$g(\boldsymbol{h}_{i,l}, \boldsymbol{m}_{i,l}) = \boldsymbol{h}_{i,l} + \boldsymbol{m}_{i,l}$$

There have been many suggestions for the form of g and f, each giving different properties to their specific architecture. For a more information on different flavours of message-passing GNN, see [79, 80].

The input to a CNN can be viewed as a highly structured graph with guaranteed, regular connectivity (boundaries notwithstanding, all pixels have a fixed number of neighbours). The extension to irregularly structured inputs is what makes GNNs more useful for molecular data. In [81], molecular fingerprints are learned using GNNs and DFT-level accuracy for molecular properties was achieved 10^5 times faster than the full quantum mechanical calculation [82].

1.2.3.5 Group Invariance and Equivariance

CNNs are usually trained using data augmentation. Specifically, input images are rotated randomly, so that the network learns to recognise an object no matter its orientation. This is necessary because while CNNs have translational invariance, they are not invariant to rotations. Model capacity is therefore wasted because for every object the CNN can recognise, it must also learn to recognise the various angles it may be presented in. Two images which are identical except for the angle of rotation can give two different outputs. This is not desirable when working with protein-ligand complexes. There is a branch of neural network design which gives this strong inductive bias to the functions learned, which relies on some mathematical concepts which I will detail here. An example of two inputs which are identical but are transformed by a roto-translation in 3D space is given in Fig 1.3.

Groups of Transformations. A group of transformations is a set of transformations $\{g_1, ..., g_n\} \in G$ equipped with a binary operation, which satisfies four properties: closure, associativity, inverse, and identity. For example, the set of translations in two dimensions equipped with the addition operator (+) forms the group $T(2)$ because it meets all four criteria:

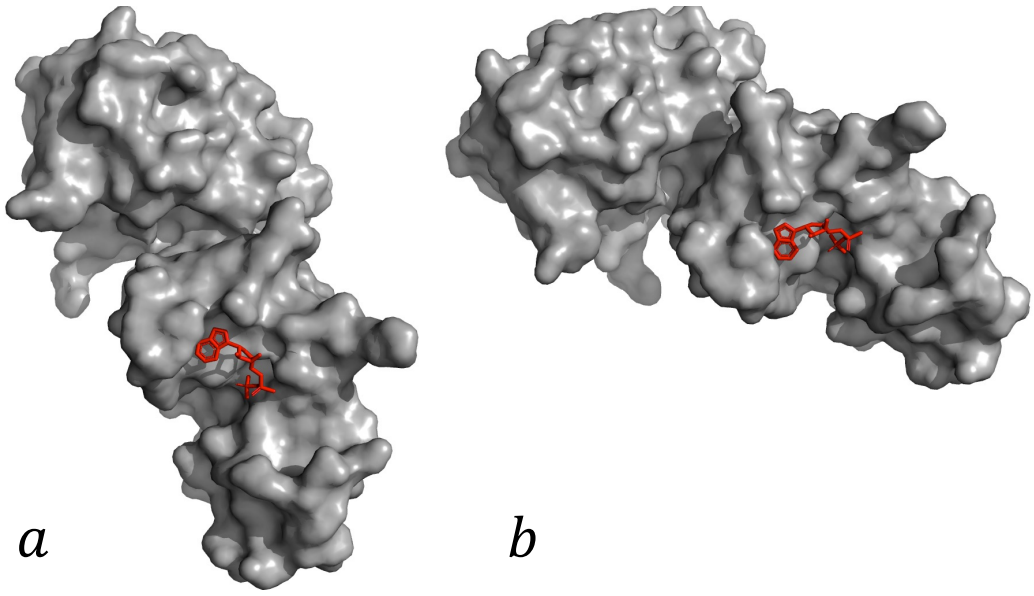


Figure 1.3: Example of the treatment of a protein-ligand complex crystal structure (PDB code 1O0H) in two different orientations by different types of neural network. The physical, chemical and biological properties of both inputs are the same (as they are the same entity), but neural networks which are not invariant to roto-translations such as standard CNNs (f_{CNN}) will assign them different predictions $y \neq z$. The $SE(3)$-invariant neural network which parameterises the function $f_{invariant}$ will give the same prediction x to both orientations.

1. **Closure under the group operation.** Using the operation on two elements of $T(2)$ results in an operation which can be expressed as a single translation, which is still a member of $T(2)$.

$$(g_1 + g_2) \in T(2) \quad \forall g_1, g_2 \in T(2)$$

2. **Associativity of the group operation.**

$$(g_1 + g_2) + g_3 = g_1 + (g_2 + g_3) \quad \forall g_1, g_2, g_3 \in T(2)$$

3. **The identity element is in the group.** This is an element $\mathbf{1}$ where $\mathbf{1}g = g \quad \forall g \in G$. For T(2), $\mathbf{1}$ is translation by zero in both dimensions, by adding $(0, 0)^T$.

$$\mathbf{1} + g = g \quad \forall g \in T(2)$$

4. **The inverses of every element are in the group.** The inverse undoes the action of a group element. For example, the inverse of an element which translates to the right by 1 is the element which translates to the left by 1. The inverse of group action g is denoted by g^{-1}, and $g^{-1}g = \mathbf{1} \quad \forall g \in G$.

$$\exists g^{-1} \in G \quad s.t. \quad g^{-1}g = \mathbf{1} \quad \forall g \in T(2)$$

A neural network for analysing a protein-ligand complex should be invariant with respect to all members of the $SE(3)$ symmetry group, which consists of all roto-translations in three dimensions.

Left-Regular Representations. A layer in a neural network is a function f from an input X to an output Y. The left-regular representation of some $g \in G$ is $\mathcal{T}_g : f(X) \to f'(X)$. Whereas g acts directly[4] on X, its *left-regular representation* $\mathcal{T}_g$ acts on functions of X to give new functions $f' : X \to Y$. A left-regular representation parameterised by a group element g acts on the domain of f by transforming its domain using g^{-1}:

$$\mathcal{T}_g[f](x) = f(g^{-1} \cdot x) \quad \implies \quad \mathcal{T}_g[f](g \cdot x) = f(g^{-1}g \cdot x) = f(x)$$

A grey-scale square image can similarly be defined as a function f from pixel coordinates $\vec{x}$ to a set of values values between 0 and 1. If, like in Fig. 1.4, we define g to be a translation rightward by 2 pixels, then when $\mathcal{T}_g$ acts on f, it produces the function that is the equivalent to f acting on x transformed by the inverse of g. We see that the function which maps $\vec{x}$ coordinates to pixel values is transformed by $\mathcal{T}_g$.

$$g \cdot \vec{x} = (x_1 + 2, x_2)$$
$$\mathcal{T}_g[f](\vec{x}) = f(x_1 - 2, x_2)$$

Group Equivariance. If the elements of G act on two homogeneous[5] spaces X and Y, a G-equivariant transformation $\phi : f(X) \to f'(Y)$ commutes with $\mathcal{T}_g$:

$$\phi(\mathcal{T}_g[f]) = \mathcal{T}_g[\phi(f)] \qquad \forall f : X \to Y \quad \forall g \in G$$

[4]This is a slight abuse of notation; the group element g acts on the vector space X via its representation $\rho(g)$, which here means a matrix representation of the group element g. In fact, ρ is a linear transformation which maintains the group structure of G, such that $\rho(g') \cdot \rho(g)(\vec{x}) = \rho(g'g)(\vec{x})$.

[5]A space X is homogenous with respect to G if and only if $\forall x, x' \in X, \exists g \in G$ such that $g \cdot x = x'$. 3D space is homogeneous with respect to $SE(3)$.

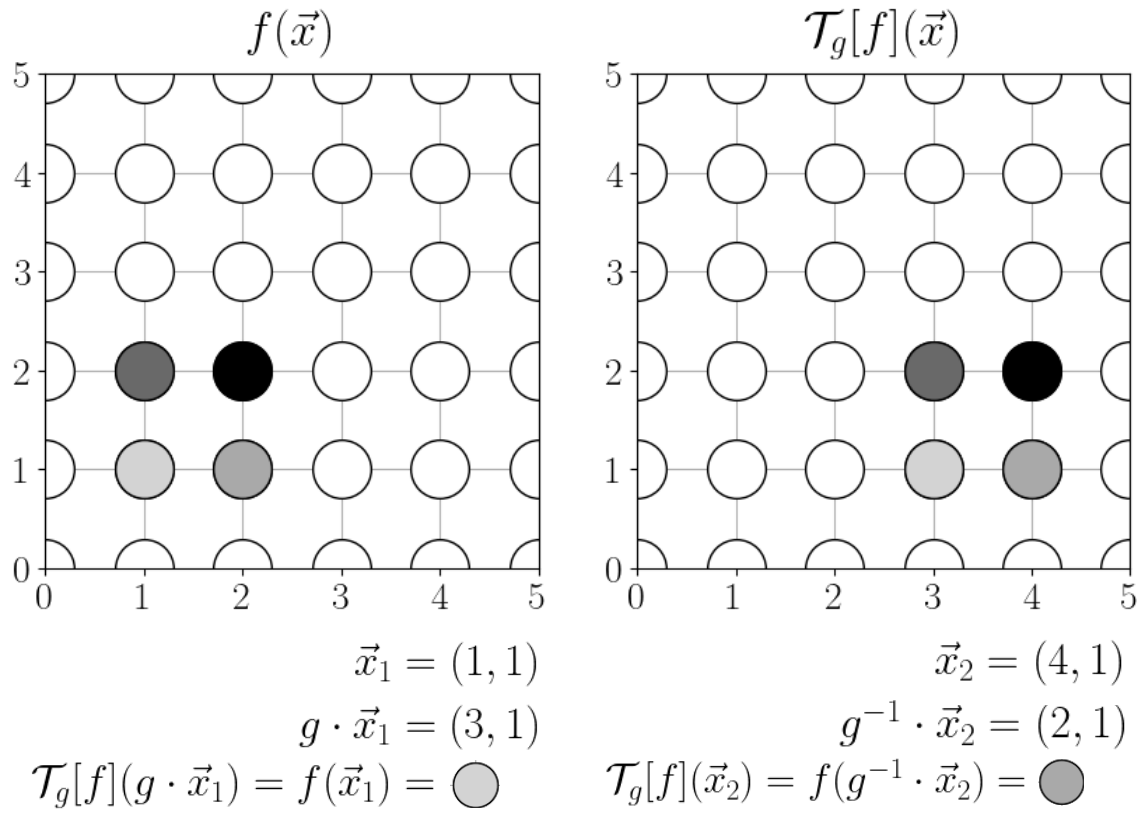


Figure 1.4: Example of the left-regular representation of an element from the $T(2)$ group of transformations. Here, g is a translation to the right by 2: $g \cdot \vec{x} = \vec{x} + (2, 0)$, so $g^{-1} \cdot \vec{x} = \vec{x} + (-2, 0)$. Two points on the domain, $\vec{x}_1$ and $\vec{x}_2$, have been chosen to highlight the relationship between the original function f and $\mathcal{T}_g[f]$ (both supported on a discrete pixel grid). As shown on the right, the output values of the new function $\mathcal{T}_g[f]$ can be found by applying the original function f to $\vec{x}$ transformed by g^{-1}.

ϕ takes as an argument a function on X and returns a function on Y, and transformations in the input function should produce an equivalent transformation in the output function. If $\mathcal{T}_g[\phi(f)] = \phi(f) \quad \forall g \in G$ then ϕ is G-invariant, meaning that elements of G acting on f have no effect on the transformation *phi*. For a neural network layer which takes an image as input and returns a feature map, this means that shifting the input upwards by 3 pixels produces an equivalent translation in the output feature map. A 2D convolutional layer can be shown to be $T(2)$-equivariant using this framework [83].

1.2.3.6 Group-Invariant Neural Networks

The difference between point clouds and graph neural networks is that nodes in a point cloud have explicit coordinates (usually on $\mathbb{R}^2$ or $\mathbb{R}^3$), whereas graphs do not have

this requirement but instead the nodes are connected explicitly by edges. There are, however, mechanisms to give connectivity to point clouds and coordinates to graph nodes, such that the distinction becomes less useful. This is a technique commonly employed when constructing graphs from points in space which have local connectivity such as from molecules, so for brevity any neural network operating on point clouds or graphs shall be called a GNN from hereonin. The three GNN architectures described here are implemented as part of the code associated with this thesis, but only the final one is used extensively.

Group invariance can be achieved by stacking group-equivariant layers under group invariant layers [84]. Many standard neural network layers are $SE(3)$-invariant, for example global pooling operations. $SE(3)$-equivariant layers which operate on graphs or point clouds can be achieved in many ways, some of which are explained below.

LieConv. To our knowledge, LieConv [85] was the first efficient $SE(3)$-invariant neural network layer which worked on point clouds, where the inputs are a set of features and their continuous coordinates $\{(\vec{x}_i \in \mathbb{R}^3, f_i)\}_{i=1}^{i=N}$. The main innovation in this work is the lifting operation, wherein inputs which are supported on $\{\vec{x}_i\}_{i=1}^{i=N}$ are 'lifted', mapping them to a (possibly infinite) set of group elements, represented by matrices. Group-equivariant operations are easier to define on the set of group elements than in Euclidean space, and indeed the authors go on to describe a group-equivariant convolution on elements of the $SE(3)$ group, parametrised by a small neural network. Their architecture achieved competitive performance against other GNNs on a range of tasks, including predicting 1D molecular properties [86] and modelling particle trajectories [85].

LieTransformer. The LieTransformer adopts the same lifting strategy as described by Finzi *et al.* in LieConv, but they opt to define a group-equivariant self-attention mechanism rather than a convolution [87]. Attention scores are analogous to weighted edges of a graph, and attention is applied to both features and the group elements representing the positions separately, before their recombination. LieTransformer was a more accurate predictor than LieConv on 8 out of 12 categories on the QM9 dataset. QM9 is a set of small molecules (supplied as graphs), labelled by 1D molecular properties including atomisation energy, geometry, HOMO/LUMO energies, dipole moment and heat capacity [88, 89], and is a common way of benchmarking GNNs. The LieTransformer was also better at predicting particle trajectories

under spring dynamics [87].

E(N)-Equivariant Graph Neural Networks. Satorras *et al.* took a different approach to achieving $E(3)$-equivariance[6]. In EGNN layers, equivariant transformations are defined directly on the node positions, removing the need for expensive lifting operations [90]. An added benefit of this is that, unlike for LieConv and LieTransformer, approximations of integrals are not required, so true equivariance to infinite groups (such as $SE(3), E(3)$) can be achieved. A small neural network was also used as an attention mechanism operating on edge features, to modulate the signals as they pass between nodes depending on their perceived importance. This lightweight layer achieved competative performance compared to other group-invariant methods on molecular properties and particle trajectories.

1.2.3.7 Attention

Sophisticated attention mechanisms have been an important recent development in neural networks, and both LieTransformer and EGNN use some form of attention mechanism. Attention weights signals by their relative importance, and it has recently been used to great effect in natural language processing tasks such as machine translation. Words in a sentence are assigned pairwise attention scores depending their relationship within the input text. These are used as weights to generate a new vector representation for each word by a linear combination of the representations of surrounding words from the text. This type of attention, between different parts of the input, is known as self-attention [91, 92]. It is motivated by the observation that words are defined implicitly by their usage in relation to other words. Self-attention was a key component in the breakthrough machine translation work in 2017 by Vaswani *et al.*, which introduced the Transformer architecture that has since proved influential in a wide and varied range of tasks [92].

In the context of EGNN layers, attention means that the message passed from one node to another is modulated by the importance of the edge. Two atoms forming a strong hydrogen bond should give each other high attention scores, and the messages passing between them should usually be given more importance than a single intermolecular carbon-carbon interaction. Likewise, attention weights should favour spatially closer atoms over distant ones. These attention scores can be used as a

[6] $SE(3)$ is a subgroup of $E(3)$.

proxy for atom-atom interaction importance. This idea is outlined in Fig. 1.5, and is explored in Chapter 4, Section 4.2.3.

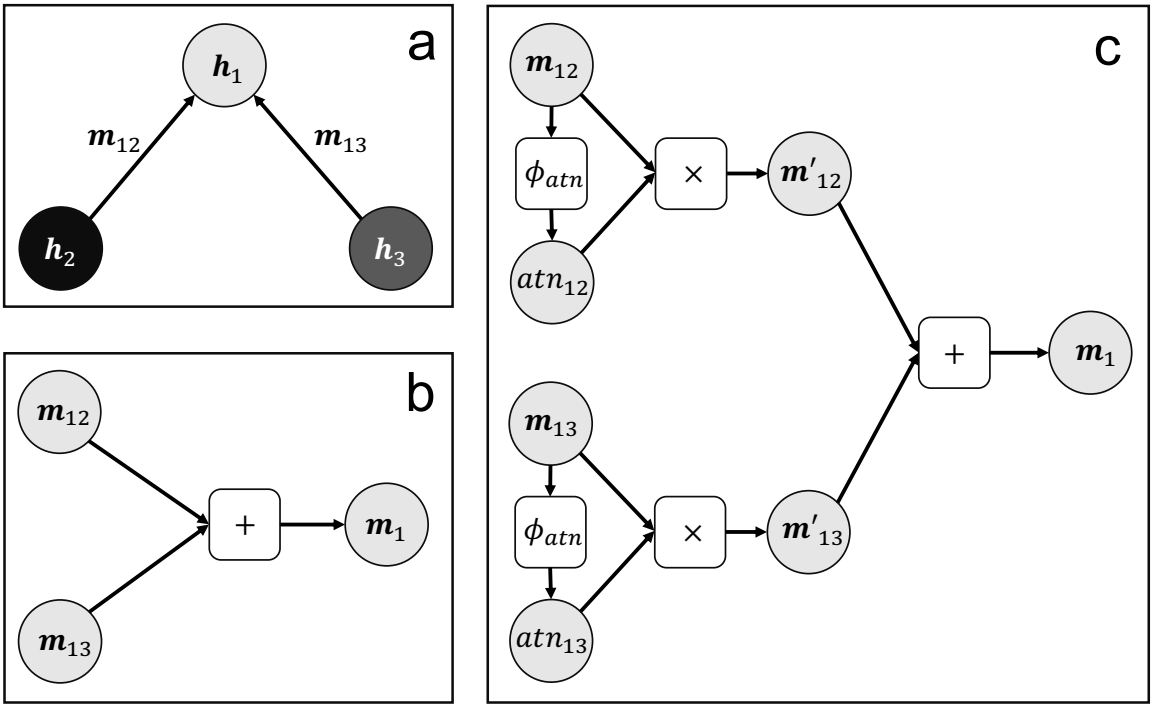


Figure 1.5: Attention in EGNN layers. (a) Message-passing diagram. Atom 1 (node 1) with node embedding $\boldsymbol{h}_1$ receives messages from atoms 2 and 3, with node embeddings $\boldsymbol{h}_2$ and $\boldsymbol{h}_3$. Each message $\boldsymbol{m}_{ij}$ is a function of the two node embeddings $\boldsymbol{h}_i, \boldsymbol{h}_j$, their positions, and the edge type. For the work in this thesis, three types are used: protein-protein, protein-ligand, and ligand-ligand edges. (b) Computational graph showing how messages in (a) are aggregated at node 1 to form a single message $\boldsymbol{m}_1$. Circles are values, and squares are operations. (c) Computational graph showing same aggregation as in (b), except the strength of the message signals is modulated using an attention mechanism. ϕ_{atn} is a small neural network, which takes the raw messages $\boldsymbol{m}_{ij}$ and returns a single value atn_{ij}. This attention weight is the weight of each message for their linear combination into $\boldsymbol{m}_1$.

1.2.3.8 Input Attribution

The decisions made by machine learning algorithms are increasingly being used in commerce, industry, science, medicine, government, and policing [93]. As their impact on daily life increases, the process by which they arrive at their decisions has come under public scrutiny [94], because decisions made by most powerful models, such as neural networks, are notoriously difficult to interrogate. The part of this interrogation concerned with assigning importance to the model inputs is called input attribution.

An example of what input attribution could look like is shown in Fig. 1.6.

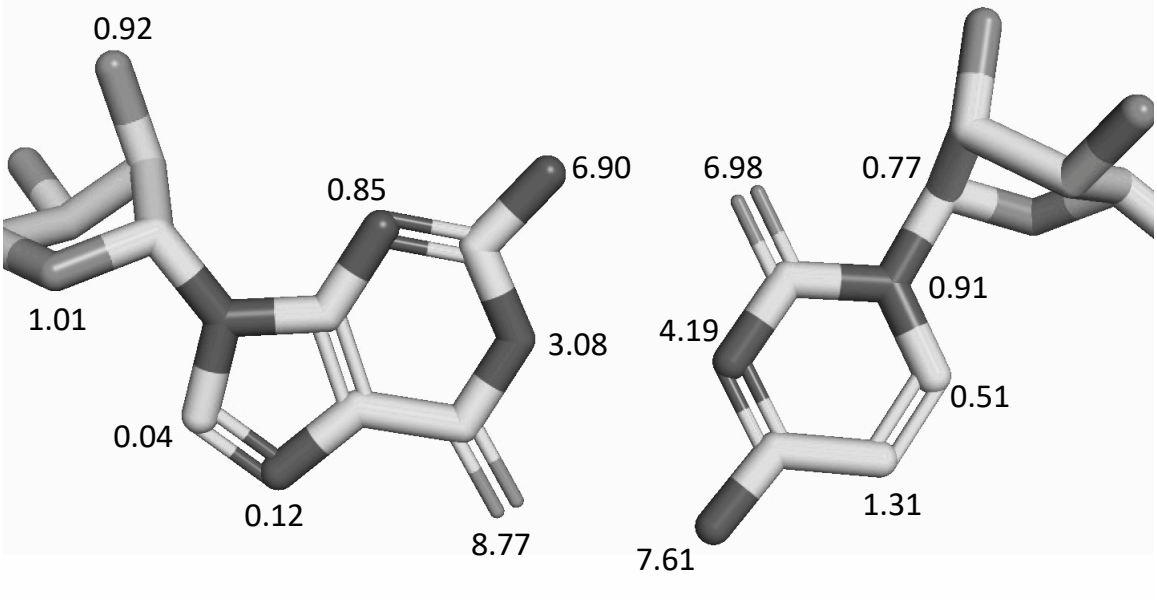


Figure 1.6: Contrived example of input attribution. The molecules shown are cytosine and guanine, and the output of the neural network could be the strength of the binding energy. More important atoms associated with hydrogen bonding are assigned a higher importance, indicated by a higher number. Here, attribution is shown for atoms only, but attribution on pairwise interactions (bonds) is also possible.

Scientists are interested in input attribution for two reasons. The first is that we want to know that predictions are being made *for the right reasons.* In this case, predictions ought to be made based on the physical interactions between the atoms in the ligand and those in the protein, as well as entropic and steric factors. This is important because the alternative, as discussed previously, is learning to separate correct from incorrect poses or high- from low-affinity binders based on other factors [63, 70, 74, 75]. These factors may not be applicable to other areas of protein-ligand space, so generalisability will be reduced, and the ability of models to predict on novel targets and ligands will be limited.

The second reason is that in the build-up to building drug-like molecules, smaller molecules are used to probe the binding site of the protein, especially in FBDD. The process by which the fragments are linked, merged or grown into larger molecules has until recently been by human inspection of bound structures, information from which is aggregated across fragments. Recent advances in fragment-to-lead algorithms are

promising [95, 96, 97], but their use is still in their infancy and they often rely on manually selecting parts of the binding site to target. Automated atom target selection based on machine learning would ensure that previous knowledge of which parts of the binding site make good targets is incorporated into fragment elaboration. Input attribution is one way to achieve this.

There are many algorithms for input attribution in CNNs which show good performance on 2D images [98, 99, 100, 101]. However, none of these account for the connectivity between parts of the input space. Class activation mapping (CAM) [102] is commonly used, but the coarsening of spatial dimensions as information flows through CNNs is a fundamental limitation. In [103], masking is used for input attribution on Gnina models. Scores are assigned to input atoms or residues based on the score difference between the prediction obtained with the full input and the prediction obtained when that particular entity is removed. Masking is used to indicate whether or not protein information used by CNNs in Chapter 2.

Unlike in CNN-based SBVS algorithms, the concept of atoms, and their interaction, can be made fundamental to GNNs in the form of vertices (atoms) and edges (bonds). This structure is maintained throughout the network until the final pooling layers, so information may be extracted from atoms and bonds at the end of the network. There exist graph-based analogues to CAM and masking [104], as well as other methods not available to CNNs such as edge attention scores, which can be used to assign importance to non-covalent interactions.

1.2.4 Thesis Structure

Chapter 2 consists of a publication on CNNs, based on the Gnina input representation for VS [63]. Bias in the DUD-E dataset is experimentally examined, and a new method for debiasing VS datasets is proposed. The networks trained on these augmented datasets are tested on a validation set constructed separately from the training set, showing modest improvements in generalisability over models trained on the original DUD-E dataset. Input attribution is used on case studies to indicate greater use of protein information by the new models for making predictions.

Chapter 3 contains implementation details for PointVS, as well as code written for a paper by Thomas E. Hadfield which explores how input attribution changes depending on dataset bias, while highlighting the need for more accurate binding data,

the preprint of which can be found at [105].

Chapter 4 is based on a publication carried out in collaboration with Lucy Vost [106]. It introduces PointVS, a GNN based on the EGNN layer for pose classification and affinity prediction. Competitive performance for both tasks is achieved, and information leakage between common train/test splits is investigated. STRIFE is a generative model defined in a recent publication by Hadfield *et al.* [107]. It uses the Hotspots API [108] to derive statistical potentials from the Cambridge Structural Database [109], identifying where to direct fragment growth. During a fragment screen, hundreds of binding events are recorded, information which the hotspot map generation procedure in STRIFE currently ignores. In our work, attribution scores for protein atoms are used to construct fragment screen-informed hotspots to be used by STRIFE, with evidence that these hotspots give better results than those from Hotspots API.

Chapter 5 contains discussion on the work carried out in the production of this thesis, as well as reflections on the state of the field and future work.

Chapter 2

Data Set Augmentation Allows Deep Learning-Based Virtual Screening to Better Generalise to Unseen Target Classes and Highlight Important Binding Interactions

Learning dataset biases have plagued deep learning models for SBVS since their inception. There are three main issues: the fact that (a) publicly available datasets are too small and noisy, (b) protein-ligand space is not sufficiently covered, and (c) there is information leakage between common training and test sets. The third point serves to mask the other two, although in some sense it is the most easily remedied by clustering by protein and ligand properties, such as pocket topology [110] or sequence similarity [111], Tanimoto similarity of ligand fingerprints [23] or 1D ligand properties [36].

The problem as highlighted by others [70, 73, 74, 75] is that 'active' and 'inactive' molecules are too easily separated by members of the function space available to neural networks, without the need to reference their interaction with any protein. This chapter is largely taken from the first publication to come from this thesis, written at a time before GNNs saw widespread use in SBVS.

It is based on Gnina, one of the first SBVS CNNs [111], and on a later variant called DenseFS [23], both of which were built to separate *actives* (molecules which have been experimentally determined to bind to a protein) from *decoys* (molecules which it is assumed do not bind). We first describe the problem with learning the

ligand biases present in the DUD-E dataset [36], including proof that removal of the ligand during training and testing has only a minimal effect on model performance. We then go on to propose a method of dataset augmentation which seeks to remove the ability for the CNNs to train by separating active from inactive ligands without considering their position relative to the protein. We use performance on a test set which is sufficiently separated from the training set, as well as input attribution, as evidence that the augmentation forces the CNNs to account for protein-ligand interactions. This work was published on July 23^{rd}, 2020 in the Journal of Chemical Information and Modeling (DOI: 10.1021/acs.jcim.0c00263).

2.1 Introduction

Current deep learning methods for structure-based virtual screening take the structures of both the protein and the ligand as input but make little or no use of the protein structure when predicting ligand binding. Here, we show how a relatively simple method of data set augmentation forces such deep learning methods to take into account information from the protein. Models trained in this way are more generalizable (make better predictions on protein/ligand complexes from a different distribution to the training data). They also assign more meaningful importance to the protein and ligand atoms involved in binding. Overall, our results show that data set augmentation can help deep learning-based virtual screening to learn physical interactions rather than data set biases.

A series of recent papers has shown that some deep learning methods designed for SBVS can accurately separate actives and decoys when given only the structure of the ligand [70, 74, 75, 73]. These results indicate that such methods are learning differences between the properties of actives and decoys, rather than the physical interactions between the receptor and the ligand. From this, it is possible to conclude both that the methods will fail to generalize well (predict on data sets far removed from the training data), and that there are significant flaws in the current training data sets and/or regimens.

The task of a VS algorithm is to distinguish between bound structures where the small molecule is an active and those where it is a decoy. A standard SBVS technique docks potential drug-like compounds into a protein target of interest and ranks them according to a physically inspired scoring function (for a review of common VS techniques, including docking, see [112]). These methods can be used to screen vast numbers of potential compounds for target interactions and are extremely cheap compared to laboratory-based experiments[113]. The accuracy of docking, however, is highly target-dependent [114]. This issue, along with the current deluge of structural data, has prompted the use of machine learning methods in SBVS and in the verification of docked structures [115, 116, 117].

Machine learning has found its way into many scientific domains, and drug discovery is no exception (a review can be found at [118]). Multiple ML-based SBVS methods have been developed. Most of these rely on 1D or 2D descriptors—that is,

they take as input a representation of the ligand as a fingerprint, [119] graph, [120, 21] or other descriptors, [121] and the protein (if present at all) as a sequence of amino acids [21]. More recently, machine learning methods which are able to capture specific spatial and chemical interactions between the protein and ligand have been introduced; the major driver for building these more complex methods is the hope that because they can learn physicochemical interactions between protein and ligand, they should be able to more accurately predict the binding of ligands and proteins far removed from the complexes on which they were trained (be more generalizable). A popular method for attempting to capture these types of spatial interactions is the CNN [122].

CNNs are a type of deep neural network which, due to their ability to capture spatial relationships between objects, are commonly used in image classification. The Gnina framework for VS [24] treats docked structures as 3D images with different channels for different types of atoms as inputs into a CNN and outputs a binary classification (active/decoy). A CNN with three hidden layers built in the Gnina framework and trained and tested on part of the DUD-E data set [36] was shown to perform better in the task of VS than using the docking score of AutoDock Vina. It had a mean test target ROC–AUC of 0.862, compared to 0.703 for AutoDock Vina (reported bye Imrie *et al.* in [23]); a ROC–AUC score is the area under the curve of the receiver–operator characteristic graph, which is a measure of the ability of a classifier to correctly rank unseen labelled data. The same input format with a deeper and more densely connected network along with protein family-specific finetuning was used to develop the DenseFS CNN, which significantly increased predictive power to a mean DUD-E target ROC–AUC of 0.917 [23]. This featurisation method has also been used to explore the nature and role of hydration in protein/ligand binding [123].

These results suggest that deep learning offers real potential in terms of SBVS. However, both CNN methods performed far worse on validation sets taken from a different database of structures (they generalize poorly). For example, in the case of DenseFS, models trained on the DUD-E data set and tested on an external data set constructed from the ChEMBL database [124] performed significantly worse by every metric (mean test target ROC–AUC and mean precision) than when the same models were tested on DUD-E.

It has been demonstrated many times that a machine learning model with good performance on held-out test data from the same distribution as the training data does not guarantee good performance on other data sets [75, 73, 23]. This lack of generalisability means that models cannot accurately classify ligands which are significantly larger or smaller than the molecules they were trained on, or ligands which are chemically distinct from the training data.

If the aim of generalizing to targets far from the training set is to be realized, it is important that SBVS learns physical interactions between the protein and the ligand, rather than properties of the ligand. One way to identify whether methods are learning such interactions is to visualize the importance of atoms or groups of atoms to the score. Hochuli *et al.* used several techniques to visualize which atoms and residues in the input to the original Gnina network were important to the classification decision [103]. One method used was input masking, where the CNN is used to score the docked complex both with and without certain atoms or groups of atoms present. These scores can then be compared in order to ascertain the importance of these atoms or groups of atoms.

There have also been attempts to quantify what precisely is being learned by CNNs for VS. It was recently shown that model performance for some methods does not suffer significantly when ligand structures in the test set are given without the protein target/receptor [73, 74]. The overall conclusion from these studies was that the methods are using very little if any information from the protein target. This finding links to the fact that in many of these data sets, receptor-free methods such as k-nearest neighbours on ligand fingerprints perform almost as well as the methods making use of the receptor [75]. It has even been shown that using a small number of simple 1D properties of ligand molecules is enough to classify accurately on the widely used DUD-E data set. The 50 decoys per active in the DUD-E data set were chosen to have the same or similar values as the active molecule for six key properties (molecular weight, clog P, number of hydrogen bond acceptors and donors, net charge, and number of rotatable bonds) whilst remaining topologically distinct; classifiers trained using just these six supposedly unbiased properties achieved good performance on held-out DUD-E targets [73]. This result indicates that differences between the properties of actives and decoys in the DUD-E data set without any protein information can be used for classification.

The high performance of CNNs for VS when only given the ligand structure and the known biases in the data set suggests that networks are not learning the physicochemical interactions between the protein and the ligand but are separating actives from decoys based on properties of the ligand alone. One well-documented source of this bias is the tendency for synthetic chemists to modify ligands with known activity against a target, such that the explored chemical space is expanded in regions that are both synthetically accessible and close to known binders [125]. This creates natural clusters of ligands labelled as actives.

A recent attempt to remedy this problem was made using both maximum unbiased validation (MUV) and asymmetric validation embedding (AVE). MUV is a measure of the clustering of actives among both actives and decoys. In [126], the distance between molecules was defined using their 2048-bit ECFP6 fingerprint, and "unbiased" training sets were generated by minimizing either MUV or AVE with a genetic algorithm. The training data were 189 targets each with at least 500 actives. Two types of test sets were generated: "standard-AUC" sets consisting of randomly selected (nontraining) ligands, and "far-AUC" sets which consist of ligands that are considered to be far (Jaccard dissimilarity between ECFP6 fingerprints ≥ 0.4) from ligands in the training set for that target. It was found that the original ("biased") models which performed well on standard-AUC often failed to perform as well on far-AUC, showing the expected lack of generalizability. However, training on the unbiased data sets led to a drop in performance on far-AUC, meaning that this type of debiasing is ineffective for improving generalizability.

All of these findings show that generating unbiased data sets is extremely difficult, which is a significant obstacle in using underdetermined systems such as CNNs for VS. One way of circumventing this problem which has not been widely explored is data set augmentation, which has the potential to combat biases in a way that does not require their identification or knowledge of their origin. In this paper, we test how CNNs use receptor information and explore methods to improve their generalizability. First, we compare the behaviour of deeper CNNs to their shallower counterparts, and find that deeper CNNs use more information from the receptor, but their classifications are still heavily reliant on the identity of the ligand. Given this result, we develop a procedure to force the CNN to learn from the protein/ligand interactions. We augment the training data set with active ligands in poses distinct from their active pose and labelled as decoys. In order to analyse the effect of this augmentation, we

look both at the ability of the method to predict and the distribution of CNN scores attained for different training conditions. On cross-validation (training and testing on different parts of the DUD-E data set), there is no significant difference in the performance between the original and new methods, but the score distributions show increased sensitivity to receptor information when training data are augmented in this way. Finally, to demonstrate the method trained with this augmented data set is learning physical interactions, we show both that the method is more generalizable, it performs significantly better on an external test set, and that atoms involved in interactions between the ligand and receptor are now important to the score.

2.2 Methods

To probe what is being learned by CNNs for and to develop methods to force the learning of physical interactions, three facets of the problem were explored. These were: the network architecture (model), the training set, and the presence or otherwise of receptor information in the test sets.

2.2.1 Datasets

All training was carried out using the Database of Useful Decoys, Extended (DUD-E) [36]. To carry out initial tests, a three-fold cross-validation strategy on DUD-E was used. External validation was carried out using the same subset of targets and ligands from the ChEMBL data set [124] as in the previous study by Imrie et al. [23].

2.2.1.1 Training set: DUD-E

The DUD-E [36] comprises more than 22,000 actives and 1,100,000 decoys across 102 protein targets. Actives are molecules which have been experimentally determined to bind in the binding pocket of a protein target, and for each active, there are 50 decoys which are assumed to be nonbinders. All actives and decoys are docked into the binding site using AutoDock Vina [27]. The decoys in DUD-E were generated to have similar physicochemical properties to their active (molecular weight, log P, number of hydrogen bond acceptors and donors, net charge, and number of rotatable bonds) but were designed to be topologically distinct.

Following the methodology of Imrie et al. [23], three folds were constructed from DUD-E where protein targets with $\geq$80% sequence identity were placed in the same fold to avoid training on structures that are similar to those in the test fold. The

details of these folds can be found Table 2.1. Intra-data set validation was carried out by training on two folds with the remaining one used for testing, giving three possible permutations.

Each DUD-E target can also be categorised into one of the following families, with numbers indicating the number of targets per category: kinase (26), protease (15), nuclear (11), G protein-coupled receptor (GPCR) (15), and other (45).

It should be noted that the decoys in DUD-E are only assumed to be inactive. This assumption is usually a good one, because the probability that any given ligand will bind to a target's binding pocket is low, but there are almost certainly ligands labelled as decoys which would, in fact, exhibit binding activity to the targets they are associated with. This is a problem for all machine learning methods in virtual screening, and more intelligent decoy selection is an area of active research [127].

2.2.1.2 Test set: ChEMBL

A set of 50 ChEMBL targets determined to be a suitable benchmark set for 2D fingerprinting methods was compiled by Heikamp and Bajorath [128]. Of these, a subset of 14 targets which had $\leq 80\%$ sequence similarity to all DUD-E targets was chosen as an external validation set in both Ragoza et al. [24] and Imrie et al. [23]. There are nine docked structures for each active, all of which are labelled as active. For each active ligand, there are ~ 100 decoy ligands, again each with nine docked structures labelled as decoys. This gives 12,275 active structures over 14 targets, with 1363 unique active ligand molecules along with 1,238,178 decoy structures.

2.2.2 CNN Architectures

Gnina [24] defines the binding site as a cube with sides of 24 Å centred on the ligand, which it splits into $48 \times 48 \times 48$ voxels, each with sides of length 0.5 Å. It also constructs 34 channels, each containing the pseudo-electron density from a different atom type and designated as either a "ligand channel" (18) or "protein channel" (16). The density at each voxel is calculated using a piecewise combination of a Gaussian and a quadratic function, both depending on the distance from the nucleus and the van der Waals radius. The combination of all 34 channels over the three dimensions gives a $48 \times 48 \times 48 \times 34$ tensor description of the input space which is used as the feature vector for the CNN. The two models described below use this featurisation method but

Fold 0	
Target	Family
abl1	Kinase
braf	Kinase
cdk2	Kinase
fak1	Kinase
igf1r	Kinase
kbcp	Kinase
mk01	Kinase
mk10	Kinase
src	Kinase
ada17	Protease
casp3	Protease
dpp4	Protease
tryb1	Protease
urok	Protease
andr	Nuclear
ppard	Nuclear
drd3	GPCR
tysy	Other
hdac8	Other
hivrt	Other
pur2	Other
aofb	Other
inha	Other
comt	Other
sahh	Other
pygm	Other
fabp4	Other
aldr	Other
fnta	Other
pa2ga	Other
xiap	Other
hmdh	Other
dyr	Other

Fold 1	
Target	Family
atk1	Kinase
atk2	Kinase
egfr	Kinase
jak2	Kinase
lck	Kinase
met	Kinase
tgfr1	Kinase
wee1	Kinase
ace	Protease
bace1	Protease
fa7	Protease
hivpr	Protease
fa10	Protease
thrb	Protease
esr1	Nuclear
esr2	Nuclear
pparg	Nuclear
thb	Nuclear
cxcr4	GPCR
aces	Other
pyrd	Other
pgh1	Other
parp1	Other
cp2c9	Other
def	Other
pnph	Other
pgh2	Other
ada	Other
cp3a4	Other
nram	Other
fkb1a	Other
ptn1	Other
hdac2	Other
grai2	Other

Fold 2	
Target	Family
csf1r	Kinase
kit	Kinase
mapk2	Kinase
mk14	Kinase
mp2k1	Kinase
plk1	Kinase
rock1	Kinase
vgfr2	Kinase
mmp13	Protease
lkha4	Protease
reni	Protease
try1	Protease
gcr	Nuclear
mcr	Nuclear
ppara	Nuclear
prgr	Nuclear
rxra	Nuclear
aa2ar	GPCR
adrb1	GPCR
adrb2	GPCR
glcm	Other
cah2	Other
grik1	Other
ital	Other
dhi1	Other
fpps	Other
pde5a	Other
nos1	Other
kif11	Other
hivint	Other
hxk4	Other
kith	Other
ampc	Other
hs90a	Other

Table 2.1: Target splits for the DUD-E dataset. Targets from different folds have $\leq$ 80% sequence similarity.

differ in the network architecture used for classification. The authors of Gnina have released a standalone version of libmolgrid [129] (the featurization part of Gnina), which can be found at Gnina.github.io/libmolgrid. My own TensorFlow version of Gnina, which uses libmolgrid, can be found at github.com/jscant/Gnina_tensorflow.

2.2.2.1 Gnina

Gnina is network with three convolutional layers (each followed by a max pool), then a fully connected final layer with a softmax activation function, giving a probability over the two categories (active/decoy). This version of the CNN [24] was the subject of all the analyses that had so far been carried out into the importance of the receptor in active/decoy classification [73, 74, 75].

2.2.2.2 DenseFS

DenseFS is a much deeper network with three sets of four densely connected convolutional layers followed by a fully connected softmax layer and cross-entropy loss [23]. This network significantly improved performance over the Gnina network on both held-out DUD-E targets and the ChEMBL set.

2.2.3 Training Sets

Training was carried out using the original DUD-E data set and three different augmented data sets:

DUD-E-Original. The original DUD-E data set, using the docked poses provided in Ragoza et al. [24].

DUD-E-Trans. DUD-E-Original augmented by three copies of each active in a random conformation, translation, and rotation, and labelled as a decoy. First the active ligand molecules are 'moved' into the centre of mass of the protein, then a distance is generated by the following method (R is the radius of the protein, taken

to be half the largest pairwise distance between protein atoms):

$$
\begin{aligned}
y_1(r) &= \exp\left(-\frac{(r)^2}{2\sigma_1^2}\right) \\
y_2(r) &= \exp\left(-\frac{(r-R)^2}{2\sigma_2^2}\right) \\
y_3(r) &= \exp\left(-\frac{(r+R)^2}{2\sigma_2^2}\right) \\
y_4(r) &= 2y_1(r) + y_2(r) + y_3(r) \\
\text{where} \quad \sigma_1 &= \frac{R}{3.5} \quad \sigma_2 = \frac{R}{2.5}
\end{aligned}
$$

$y_4(r)$ is calculated for a range of 1,000 values of r in the interval (-1.3R, 1.3R). The set of values for $y_4(r)$ is then divided by their total sum, making a discrete distribution over r from which the distance to move the molecule in the direction of a random unit vector is drawn. The OpenEye Toolkit [130] is used to randomly rotate and then randomize the conformation of each translated molecule. These new active/target structures are labelled as decoys for training. See Fig. 2.2 for a visual example of the result of the protocol, and Fig. 2.1 for a plot of probability density vs. fraction of protein radius resulting from the above algorithm.

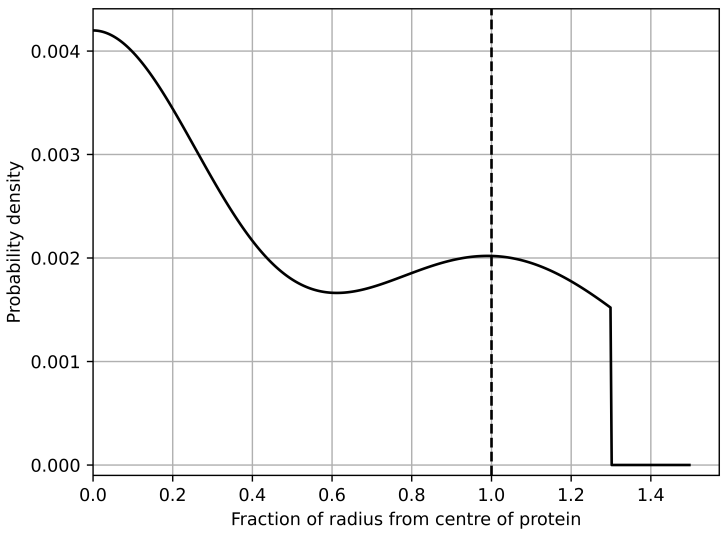


Figure 2.1: Distribution from which distances from the centre of the protein are drawn. R is half the largest pairwise distance between protein atoms. The random variable drawn from dotted line would cause the ligand to be moved to the edge of the protein, if the protein were a perfect sphere.

DUD-E-Redocked. DUD-E-Original augmented by taking each active and re-docking it into the binding pocket with AutoDock Vina to generate up to 20 poses. The three highest ranked (lowest energy) poses at least 5 Å root-mean-squared distance from the active pose were labeled as decoys and used in training.

DUD-E-Hybrid. Both augmentations from DUD-E-Trans and DUD-E-Redocked were added to the training set, giving a total of six extra decoys per active.

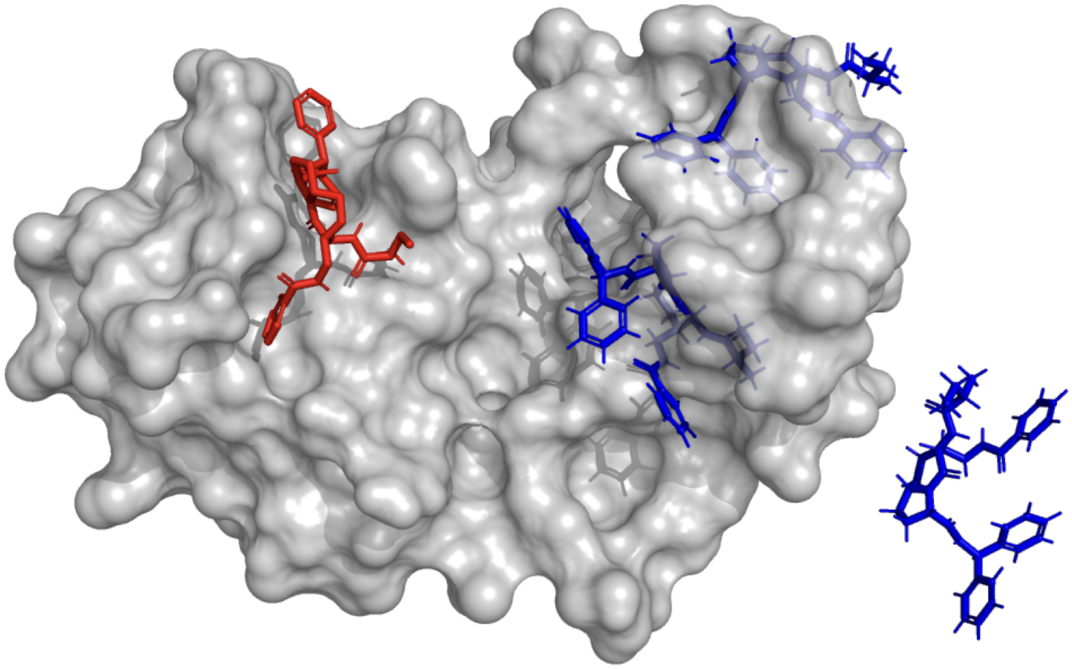


Figure 2.2: Example of random translations and conformations (shown in blue) of an active molecule (original pose in red) for a protein target (grey), included in the DUD-E-Trans dataset. Blue poses are labelled as decoys in a DUD-E-Trans training set, with red remaining as an active. DUD-E target: XIAP; ligand: CHEMBL584393.

2.2.4 Experiments

From hereon in, baseline will refer to models with the Gnina architecture trained on all or part of DUD-E-Original. OriginalFS, TransFS, RedockedFS, and HybridFS will refer to classifiers with the DenseFS architecture trained on all or part of DUD-E-Original, DUD-E-Trans, DUD-E-Redocked, and DUD-E-Hybrid, respectively.

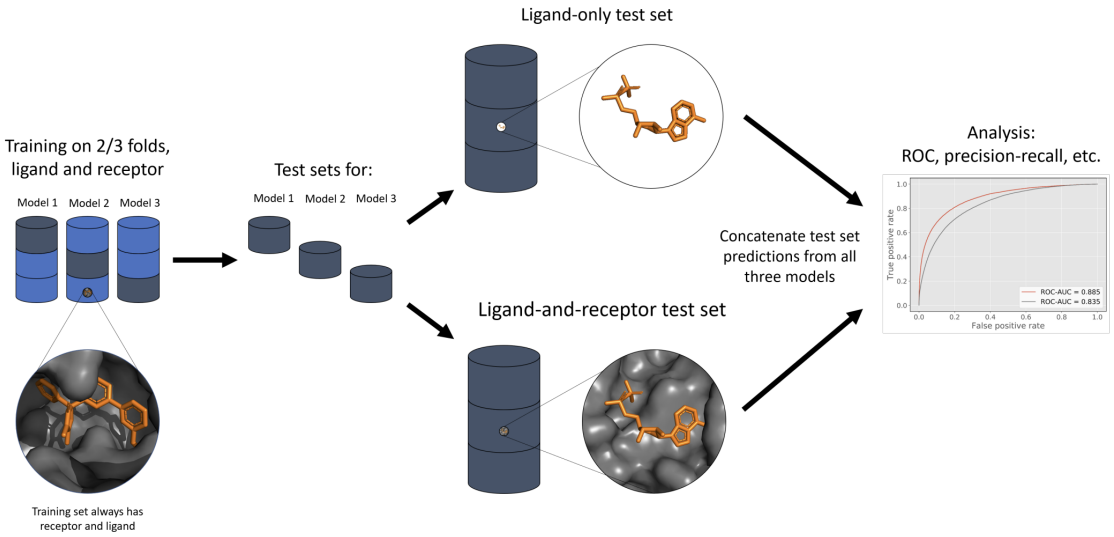


Figure 2.3: Protocol for training on DUD-E and the Ligand-and-receptor and Ligand-only tests. Three models are trained on 2 folds each (light blue), and tested on both protein-ligand and ligand-only versions of the final fold (dark blue). Test predictions from the three models are concatenated for analysis.

2.2.4.1 Training protocol

Three separate CNNs (one for each fold) were trained using each of these five CNN models, giving a total of 15 trained models. Training was conducted with a batch size of 16 for 25,000 iterations, and the number of actives and decoys in each batch was even. (18) The optimizer used was stochastic gradient descent, with a base learning rate of 0.01, an inverse learning rate power decay value of 1, a gamma value of 0.001, a weight decay of 0.001, and a momentum of 0.9.

2.2.4.2 Ligand-and-Receptor and Ligand-Only Tests

The two tests described in this section (ligand-and-receptor and ligand-only) were constructed according to the same protocol, the only difference being the presence or absence of the receptor in the test set structures. As described above and following Imrie et al. [23], three folds were constructed from the 102 DUD-E targets (Table 2.1). A model was trained on two of the three folds, with the final one reserved for testing. The modifications (translations, rotations and conformational changes) used on some of the training sets were not applied to any test set structures. Test predictions from all three models were concatenated for easier visualization of results. This protocol is depicted in Figure 2.3.

Ligand and Receptor Test. Once the models were trained according to the above protocol, they were tested on each of the test folds with no changes. This was the ligand-and-receptor test.

Ligand-Only Test. A receptor-free version of each test fold was generated. For every case (actives and decoys), the protein was removed from the structure file leaving only the docked ligand in the pose it has in the protein-binding pocket. This was the ligand-only test.

2.2.4.3 ChEMBL Validation

In order to investigate the effect of our training data augmentation on how CNNs generalise to data from a different distribution to the training set, three CNNs were trained on the entire DUD-E data set (or an augmented version) with different random seeds. These CNNs were then used to classify actives and decoys in our ChEMBL validation set. Following Imrie et al. [23], the score assigned to each structure was the mean of the three CNN scores; the score assigned to a ligand with a target was the mean of the scores given to the top five (of the nine) structures. This system is illustrated in Fig. 2.4. All training was conducted with the same hyperparameters, as outlined above.

In Imrie et al., models were first trained on DUD-E in the same way as described in the Methods section. However, the authors also made use of "finetuning," a process of further tweaking all or part of the network for specific tasks. In this instance, copies of the initially trained networks were trained on different target families in DUD-E, and then the test split was filtered according to family such that the network used to classify on each family would be that which was finetuned on the training data from that family. This was reported to greatly improve discriminative power. We did not to employ that method as the aim is only to test whether augmentation improves generalizability of the model.

2.2.4.4 Masking

Classifications from neural networks are notoriously difficult to interrogate. The masking method of Hochuli *et al.* attempts to obtain the contributions from different residues in the binding site and different atoms of the ligand [103]. In short, a trained model is given a bound structure and then the same structure with different ligand atoms or receptor residues removed (the structure is "masked"). The difference in

	Model 1	Model 2	Model 3	Pose mean
Docked pose 1	0.93	0.91	0.95	0.93
Docked pose 2	0.99	0.90	0.87	0.92
Docked pose 3	0.92	0.94	0.90	0.92
Docked pose 4	0.83	0.85	0.78	0.82
Docked pose 5	0.84	0.71	0.88	0.81
Docked pose 6	0.78	0.77	0.76	0.77
Docked pose 7	0.74	0.72	0.64	0.70
Docked pose 8	0.66	0.66	0.66	0.66
Docked pose 9	0.54	0.60	0.68	0.61

Ligand score
0.88

Figure 2.4: How a final CNN score is arrived upon given three scores for nine different poses of the same ligand for each target. Each pose is scored by three different models trained in the same way from a different starting configuration; the mean of these three scores is taken, giving nine scores (one per pose), the top five of which are averaged to give the final score.

the CNN score between the full structure and the masked structure is taken as the contribution of whichever entity was removed (see Fig. 2.8). This difference is then converted to a percentage of the original CNN score. The sum of all of the individual atomic contributions need not, in general, add to 100% because the relationship between the input features and the CNN score is nonlinear.

In the original paper (and accompanying software), only residue contributions from the receptor were calculated. An attempt at assigning importance to individual atoms was made using quantum-mechanical methods [131], but no methods for atom-level resolution for neural network-based virtual screening have been published. Here, we modified the software to mask individual atoms of the protein in order to gain a higher resolution image. The PDB structures used for masking (1O0H [132] and 1W4O [133]) were chosen to be the same as those used for masking by Hochuli *et al.* in ref [103] for ease of comparison. Neither of the PDB structures contains protein/ligand combinations found in DUD-E.

The input space only takes into account the location of atoms and contains no

concept of functional groups, pharmacophores, or bonding. Masking individual atoms does not take into account the nonlinear effects that groups of atoms can have on bonding, so will not pick up on structures whereby multiple protein atoms in certain geometries will have a stronger effect on binding than the sum of their individual masking scores. Masking is only used here as evidence that CNNs trained using augmented data assign more importance to areas significant to bonding than CNNs trained on DUD-E-Original.

2.2.4.5 Methods of Assessment

Two metrics were used to measure the discriminative power of the classifiers, the area under the curve of the receiver-operator characteristic (ROC-AUC) and the area under the of the precision-recall curve (PR-AUC). While the ROC–AUC usually lies between 0.5 and 1.0 (for random and perfect classifiers), the PR–AUC will usually lie between $n_{\text{active}}/n_{\text{total}}$ and 1.0, where n_{active} is the number of actives in the test set, and n_{total} is the total test set size. These lower bounds (which represent the performance of a random classifier) are $\sim$0.02 and $\sim$0.01 for the DUD-E test sets and ChEMBL validation sets, respectively.

The use of ROC–AUC as a metric for model performance on highly imbalanced data sets such as DUD-E (50:1 actives/decoys) or the ChEMBL set used here (100:1) has been shown to be less useful than PR–AUC [134]. ROC–AUC is widely reported for classification algorithms and is included here for ease of comparison with other work.

2.3 Results

Only results using models trained on DUD-E-Original (OriginalFS), DUD-E-Trans (TransFS), and DUD-E-Redocked (RedockedFS) will be discussed here. The results for DenseFS architectures trained on DUD-E-Hybrid (HybridFS) showed similar trends to TransFS. For more results using HybridFS, see the supporting information for [63].

2.3.1 Ligand and Receptor Test and Ligand-Only Test

The PR (left) and ROC curves (right) for the ligand-and-receptor test (green) and ligand-only test (purple) for the Baseline, OriginalFS, and TransFS models are shown in Figure 2.5.

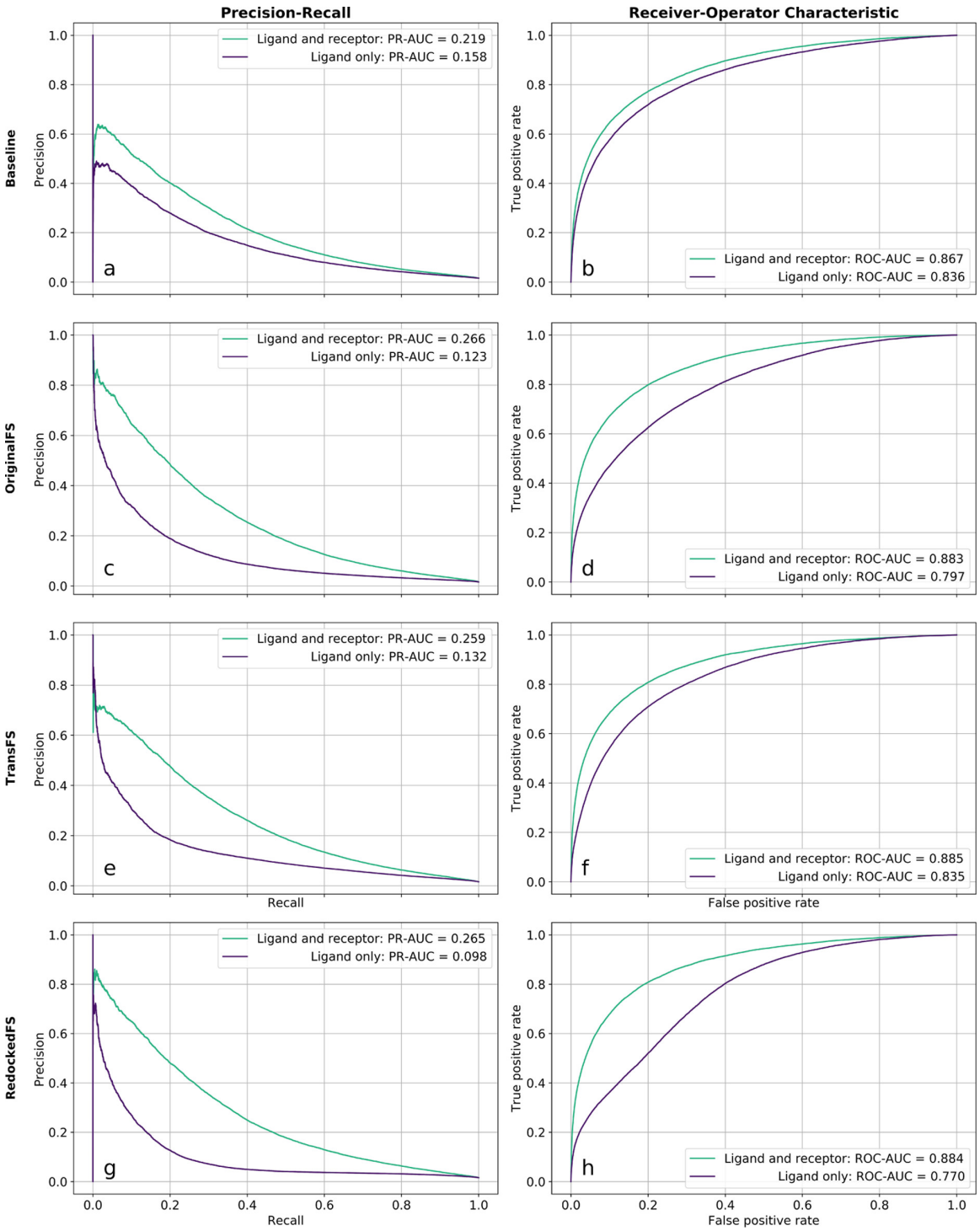


Figure 2.5: Removing receptor information does not cause total collapse in discriminative power for Gnina or DenseFS-based CNNs. PR and ROC curves for the ligand-and-receptor and ligand-only tests. Models shown are Baseline (a,b), OriginalFS (c,d), TransFS (e,f), and RedockedFS (g,h). Green lines indicate held-out DUD-E target test sets with both receptor and ligand information (ligand-and-receptor test); purple lines are the same set but with only ligand information given (ligand-only test).

2.3.1.1 Performance of Baseline

Graphs **a** and **b** in Figure 2.5 show the PR and ROC curves for Baseline models tested both with and without the receptor present. The difference in the ROC–AUC between the ligand-and-receptor and ligand-only tests is just 0.031 (3.5%, 0.867-0.836). This agrees with the previous findings by Chen *et al.* that performance is hardly affected by the removal of the receptor [75]. The PR-AUC drops by 0.061 (27.9%, 0.219 - 0.158) which is a larger change, but both metrics reflect that a large percentage of predictive ability is still retained in the ligand-only test. This result indicates that the Baseline model is not relying significantly on information from the receptor channels for its classifications.

2.3.1.2 Performance of OriginalFS

Graphs **c** and **d** of Figure 2.5 show the PR and ROC curves for the ligand-and-receptor and ligand-only tests using OriginalFS. The drop in performance when receptor information is removed is far larger than the drop seen for the Baseline. The ROC–AUC drops by 0.086 (9.7%, 0.883-0.797), and the PR–AUC decreases by 0.143 (53.8%, 0.266–0.123). The larger reduction in predictive ability when the receptor is not present indicates that the more expressive DenseFS CNN architecture uses receptor information more in its classifications than Gnina. However, there is still significant predictive ability in the ligand-only test. OriginalFS is still able to relatively accurately separate actives from decoys using only ligand information.

In an effort to force the DenseFS CNN to learn more from the receptor and how it interacts with the ligand, we trained TransFS using our augmented DUD-E set and DUD-E-Trans. The network therefore sees examples of active ligands in non-physical positions and conformations marked as decoys as well as the correct active poses.

2.3.1.3 Performance of TransFS

Plots **e** and **f** show the PR and ROC curves for the ligand-and-receptor and ligand-only tests using TransFS. The performance is very similar to OriginalFS; the difference in ROC–AUC is 0.050 (5.6%, 0.885–0.835) and PR–AUC drops by 0.127 (49.0%, 0.259–0.132). This is surprising as the TransFS models trained with DUD-E-Trans have seen examples of active ligands in non-physical structures labelled as decoys,

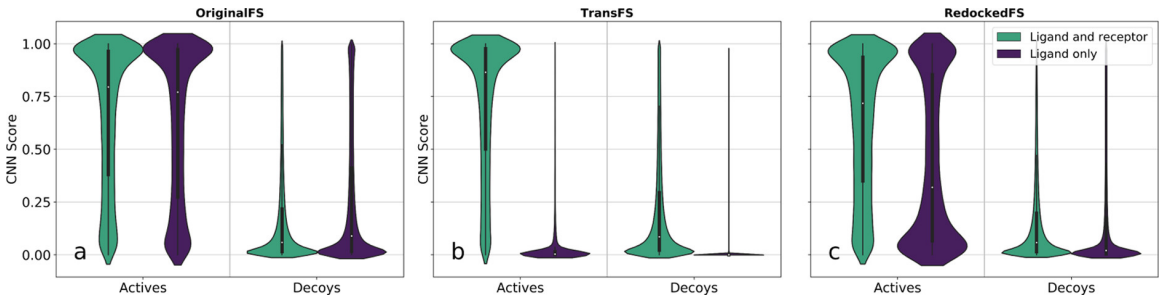


Figure 2.6: CNN score distributions reveal presence (OriginalFS) and absence (TransFS and RedockedFS) of discriminative power when receptor information is removed in ligand-and-receptor and ligand-only tests. Violin plots of test set scores given to actives and decoys by OriginalFS, TransFS, and RedockedFS CNNs. Score distributions given with ligand and receptor information present shown in green and with ligand information only shown in purple. Plots a–c show distributions for OriginalFS, TransFS, and RedockedFS CNNs, respectively.

which should remove the ability of the network to classify based on the ligand fingerprint alone.

For OriginalFs and TransFS, there is a significant but similar drop in performance when the receptor is removed, and a larger drop for RedockedFS. This suggests that using the DUD-E-Redocked augmented data set is causing greater use of the receptor in classification, whereas using DUD-E-Trans is not. In order to explore what, if any, influence training using DUD-E-Trans has, we examined the distributions of the raw CNN scores.

The effects of training set on raw CNN scores for actives and decoys in the ligand-only test are shown in Fig. 2.6. The width of the plots of a particular CNN score is proportional to the relative density of structures assigned that score. Plots are split by training set, whether structures are actives or decoys, and whether a receptor was present (green) or not (purple) in the test set.

2.3.1.4 Score Distributions for OriginalFS

Fig. 2.6a shows the distributions of pose scores given to actives and decoys in the ligand-and-receptor (green) and ligand-only (purple) tests by OriginalFS CNNs. Neither actives nor decoys exhibit a significant difference in pose score distribution between the ligand-and-receptor and ligand-only tests, that is, the green (ligand-and-receptor) and purple (ligand-only) plots for actives are almost identical. Compared to

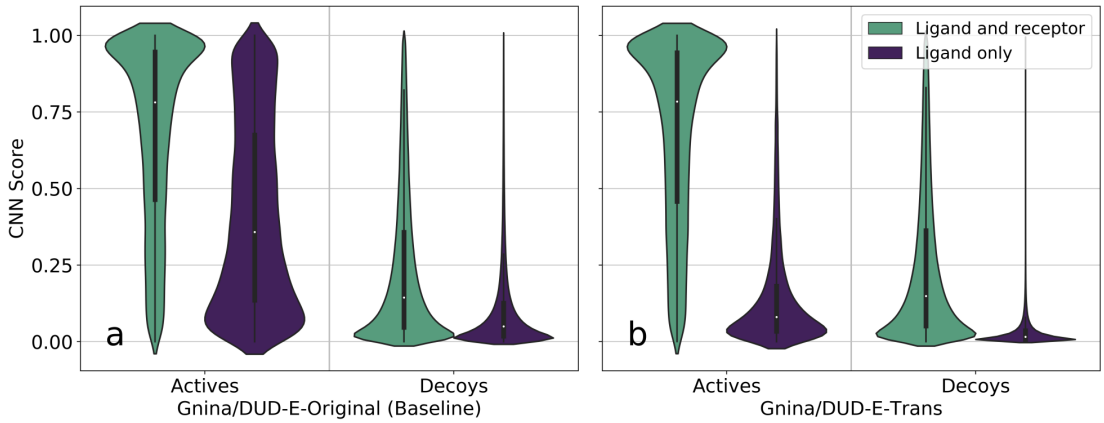


Figure 2.7: Violin plots of scores given to actives and decoys by Gnina models trained with DUD-E-Original (Baseline) and DUD-E-Trans. Score distributions for test sets with ligand and receptor information present are shown in green; test sets with ligand information only are shown in purple. Plots a and b show distributions for models trained using DUD-E Original and DUD-E-Trans respectively.

Baseline (Fig. 2.7a), decoys in the ligand-and-receptor test are skewed more toward zero, but actives in the ligand-only test are in fact skewed more toward higher scores, showing that using the deeper network alone does not guarantee the use of receptor information.

2.3.1.5 Score Distributions for TransFS

Fig. 2.6b shows the same tests as Fig. 2.6a, but instead using TransFS. Compared with OriginalFS, the score distributions when the receptor and protein (ligand-and-receptor) are present do not change significantly. However, the distributions for the ligand-only tests collapse to being highly clustered near zero - for actives, the median score goes from 0.864 to 0.0292 when the receptor is removed, and for decoys the median drops from 0.0855 to 0.000132. These results show that receptor information is being used to generate a significant proportion of the score of the CNN for TransFS.

2.3.1.6 Score Distributions for RedockedFS

The score distributions for RedockedFS networks are shown in Fig. 2.6c. It can be seen that unlike TransFS networks, the ligand-only scores for actives do not collapse to a near-zero median: the score change for actives upon removal of the receptor is from 0.718 to 0.320. For decoys, the score drops from 0.0579 to 0.0203. This shows that there is an increased need for receptor information for assigning a high CNN

score, but the effect is less pronounced than for TransFS. This could be because some augmented decoys that are generated to produce DUD-E-Redocked may be closer to the ground truth than the poses assumed to be correct.

However, this is not evidence that it is being used in the way that we desire (to pick out physicochemical interactions). In order to test whether receptor information is now being used in this way, we next test performance on an external validation set for generalizability. As described in the Introduction section, the ability to better classify structures from a different distribution to the training data would imply that general physical rules have been learned.

2.3.2 ChEMBL Validation

In order to examine how well the OriginalFS and TransFS CNNs generalize, they were used to predict on the ChEMBL validation set (see Section 2.2.4.3). The results for each target[1] in the ChEMBL set are given in Table 2.2.

TransFS performs similar to or better than OriginalFS for 13 out of 14 targets, suggesting that augmentation causes models trained on one data set to generalize better and have increased predictive power on a different data set. Spearman's rank correlation coefficients of all CNN scores for a target show strong but not perfect correlations, with none above 0.95. The two-tailed p-values are all below 64-bit floating point machine precision ($1.1 \times 10^{\check{}16}$). This shows that there is a statistically significant difference in the ranking of molecules by TransFS as compared to OriginalFS.

Training with DUD-E-Trans (TransFS) rather than DUD-E-Original (OriginalFS) tended to give larger improvements in PR–AUC scores for targets from families which do not feature heavily in the DUD-E set such as proteases (see Methods for an overview and Table S1 in the Supporting Information for a detailed breakdown of DUD-E target families). The largest performance increases were seen for proteins in the "other" category — containing targets in families such as "phosphodiesterase 4A" and "thymidylate synthase" — which are less well represented in DUD-E. This larger difference in performance for the more distinct targets is to be expected as ligands

[1]Since the release of the ChEMBL validation set [128], multiple other publications ([23, 135, 136]) have used it with the original ChEMBL target codes; while the target codes have changed in later versions of ChEMBL, we continue to use the original identification scheme here. The version of ChEMBL used to compile the original dataset can be found at https://ftp.ebi.ac.uk/pub/databases/chembl/ChEMBLdb/releases/chembl_07, and all target codes in this work correspond to that version of ChEMBL.

Chembl Target	Family	OriginalFS	TransFS	RedockedFS
10752	Kinase	**0.289**	0.272	0.270
12670	Kinase	0.363	**0.374**	0.350
20014	Kinase	0.573	**0.580**	0.566
10378	Protease	0.047	**0.052**	0.032
10498	Protease	0.056	**0.078**	0.073
11534	Protease	0.062	0.095	**0.100**
219	GPCR	0.116	**0.153**	0.069
11279	GPCR	0.039	0.039	**0.040**
11631	GPCR	0.245	**0.264**	0.191
12968	GPCR	0.009	**0.011**	**0.011**
28	Other (thymidylate synthase)	0.286	**0.363**	0.297
276	Other (phosphodiesterase 4A)	0.318	**0.415**	0.269
11359	Other (phosphodiesterase 4D)	0.214	**0.270**	0.133
18061	Other (sodium channel protein type IX α-subunit)	0.031	0.031	**0.031**

Table 2.2: PR–AUC values for the 14 targets in the ChEMBL validation set for OriginalFS, TransFS, and RedockedFS CNNs. Highest scores are highlighted in bold.

which are active for one member of a protein family are likely to be active for others, so learning the fingerprint profile of ligands likely to bind to one member of a family will help performance on external validation sets which contain that family. This can be seen here in the performance on kinase targets, which is extremely close between the two different training regimens. There are 26 kinase targets in the DUD-E training set, and ligands which are active for one kinase are much more likely to be active for another. In our case, this means that OriginalFS performs comparably on kinases to TransFS, and in the case of the ChEMBL kinase target with ID 10752, OriginalFS has a PR–AUC that is 0.017 (5.9%) larger than the PR–AUC achieved by TransFS.

The ratio of actives to decoys in the ChEMBL validation set is 1:100; this level of class imbalance means that the false positive rate is not increased greatly by erroneously assigning more active labels to decoy ligands because the denominator is large and dominated by the number of true negatives, so a high ROC–AUC can mask low precision. This is not the case for PR–AUC, which focuses on performance on the minority active class. When there is a large class imbalance, precision is greatly reduced when models assign minority labels to the majority class. These facts show that fewer decoys are erroneously labelled as active by CNNs trained on the augmented versions of DUD-E.

In general, the performance of RedockedFS on the ChEMBL validation set is worse than for TransFS. This difference is only small for kinases, proteases, and GPCRs, but significant in three of the four targets in the "other" category, with RedockedFS performing significantly worse than OriginalFS on these targets. One possible explanation is that docking does not reproduce the exact ground truth, so producing decoys by redocking actives may in fact cause the data set to contain poses which are close to the ground truth to be labelled as decoys. This added noise results in significantly worse performance, especially on targets which are unlike targets in the training set (DUD-E). When we translate into physically nonsensical positions and conformations to produce decoys as we did for TransFS, we can be sure that these are accurately labelled as poses which are not seen in nature, so such confounding noise cannot exist.

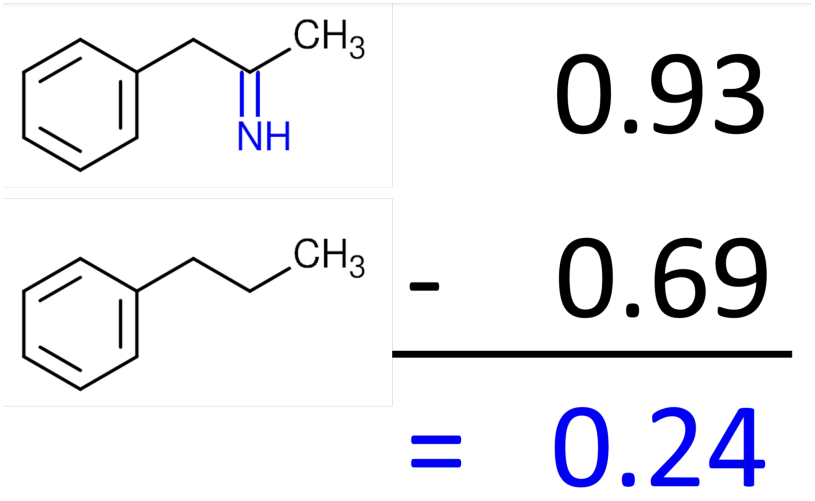


Figure 2.8: Atomistic masking. The contribution of each atom is given by the difference in scores when the atom is present compared to when it is not.

Being better able to generalize to an external validation set, especially on targets from families not in the training set, suggests that more general rules of physical interactions are being learned. In order to look for more direct evidence of this, we used input masking to find which areas of the ligand and protein each CNN assigned importance to when making a classification.

2.3.3 Masking

Masking can be used to identify which parts of the feature vector are important in the classification given by a CNN, by "covering" parts of the input and comparing

the score to the "uncovered" original (see Fig. 2.8). Here, we have used it to check the contribution of each individual atom in both the ligand and receptor. A CNN which learns physical interactions should give high (positive) masking scores to atoms involved in favourable interactions such as hydrogen bonding, and negative scores to atoms involved in unfavourable interactions such as steric clashes. Masking was performed as outlined in the methods on PDB structures 1O0H and 1W4O, two of the structures used for masking in Hochuli et al. (21) (Figure 2.9), neither of which are present in DUD-E.

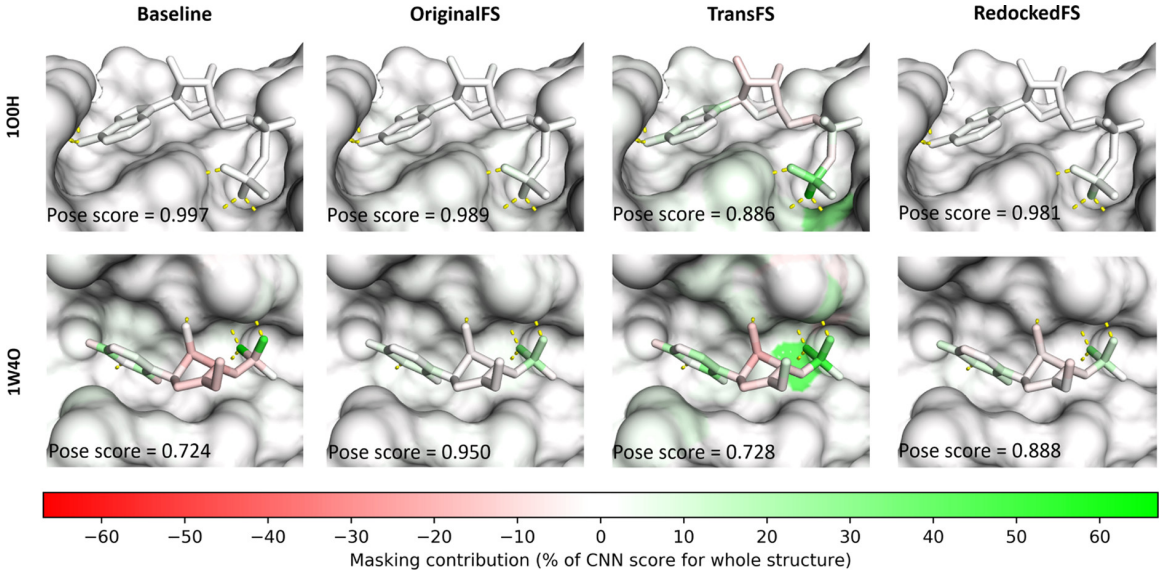


Figure 2.9: Masking using TransFS CNNs highlights important interactions between the ligand and protein. Results of masking for bound structures with PDB IDs 1O0H (top) and 1W4O (bottom). Hydrogen bonds (defined as donor/acceptor pairs within 3.5 Å of one another) are shown as yellow dashed lines. The pose scores from different networks and training sets are shown in the bottom left corners of each image. Color scale shows colors assigned for different masking scores as a percentage of the pose score of the original unmasked structure.

Baseline (left). There is no contribution displayed by any of the atoms of the receptor for either 1O0H or 1W4O. The highest ligand atom score is 2.4% of the overall CNN score for 1O0H and the two highest scoring atoms for 1W4O are oxygens which appear to be involved in hydrogen bonding to receptor atoms (66.3 and 50.5%).

OriginalFS (middle-left). A small contribution to the score can be seen for the terminal phosphate of the ligand in 1O0H potentially related to hydrogen bonding to the receptor, although this is not matched by coloring in the receptor. The largest

contributions for 1O0H are 1.1% (receptor) and 5.6% (ligand). There are larger ligand contributions for 1W4O (max. 34.5%), but again these are not related to interactions sites on the receptor (max. 1.7%).

TransFS (middle-right). There are stronger signals from atoms in the binding sites of the receptors of both structures. The strongest signals from both the ligand and receptor in 1O0H are for atoms involved in hydrogen bonds between the receptor and terminal ligand phosphate group. The largest contributions are 28.6% (receptor, a nitrogen) and 59.8% (a hydrogen-bonding oxygen on the phosphate group in the ligand). The phosphorous atom in terminal group of the ligand is assigned a score of 35.3%. Similar interactions are picked up for 1W4O, again with atoms involved in hydrogen bonds between the ligand and receptor being highlighted as important to the CNN score. Here, the largest contributions are 38.5% (receptor) and 54.9% (ligand).

RedockedFS (right). There is no signal above 1% from any protein atoms. The ligand atoms which are given high masking scores by RedockedFS are the same ones highlighted by the other three models. It can be seen that, at least for the two structures shown, RedockedFS does not assign importance to receptor atoms associated with hydrogen bonding. The results are similar to those obtained using OriginalFS.

These results suggest that training on DUD-E-Trans (TransFS) causes CNNs to score receptor and ligand atoms involved in hydrogen bonding, an effect which has not been shown before. They also suggest that networks trained in this way are learning physical interactions, a requirement for a more generalizable classifier.

2.4 Conclusions

In this chapter, we have described a method of training data augmentation that improves the generalisability of a deep learning method for structure-based virtual screening. Our augmentation consists of adding three copies of each active in a random conformation, randomly rotated and translated, and labelled as decoys for training. Networks trained with these augmented data also have higher scores associated with receptor and ligand atoms that participate in favourable interactions. Our simple augmentation procedure could be configured for use in many other machine learning methods for ligand-binding prediction. The arbitrary choice of using

three translated decoys for each active could also be further tuned, for example, using cross-validation on a test set external to the training data.

It has been almost 30 months since the publication of the paper which constitutes the majority of this chapter. The field has largely moved on from CNNs to GNNs and geometric deep learning has made important advances [137]. The work carried out since then as part of this thesis reflects this, but there are some other criticisms and comments of the training and testing protocols and the underlying philosophy of the work in this chapter.

The diagnosis that CNNs trained on the DUD-E dataset will learn dataset biases, rather than physical binding interaction rules, is correct, as is the assertion that this can be mitigated against by dataset augmentation. This is an idea that has been used in a subsequent paper by the authors of Gnina, albeit with more sophistication: binding datasets are augmented based on uncertainty in model predictions [111]. The use of input attribution to verify how decisions are made is also vital to ensuring models are not 'cheating', and that they will generalise to unseen ligands and proteins.

The major criticism of the training/testing protocol is that, for all models, pose validation is conflated with faux binary choice of 'would bind' and 'would not bind'. Using the original DUD-E dataset for training, the correct label for an active docked into an energetically unfavourable position is ambiguous. Even for TransFS, this ambiguity is still present for a very weak binder in the 'correct' pose. The assumption in DUD-E that non-binders would not bind is not absolute: the 9,219 verified 'non-binders' in DUD-E are only verified up to 30 μM, and the rest are only presumed inactive. Any false labelling introduces noise, and conflates the two questions being asked at the same time of every pose.

This mixing of two types of query, that of the identity of the binder as well as its pose, also has implications for the ChEMBL validation set. Here, different poses for the same active molecule were scored, and averages taken - but there is no guarantee that those poses were close to the native pose. This means that networks which can correctly discriminate an active molecule in a good vs bad pose may score the ensemble as inactive, if the ensemble contains mostly bad poses. This would harm the reported PR-AUC and ROC-AUCs.

In Chapter 4, we discuss these flaws further and attempt to address them.

Chapter 3

PointVS - Building Robust, Reusable and Adaptable Code for Virtual Screening

The code associated with the paper that constitutes most of Chapter 4 started as a 'toy', and is now the basis upon which several members of the Oxford Protein Informatics Group have built projects. Some of these projects I did not contribute to beyond the code base I had written for my own purposes; for two I had a more hands-on role. The first of these is code for the fragment screening data-driven extraction of protein hotspot maps, which are be covered in Chapter 4. The second is a paper by Thomas Hadfield, which examines what is learned by various ML-based scoring functions, as well as their robustness to noise. The preprint can be found at [105], and I describe the main results in Section 3.6. I shall discuss the development and implementation of the code in this chapter.

Several engineering and design choices have been made in the code as the scope has widened and user-base expanded from its original sole developer. Installation has been made simple on Linux and MacOS (both Intel and Apple Silicon[1]), and the modular nature of the core machine learning code means that implementing other architectures is relatively easy. Generating datasets from standard molecular input formats has been automated to some extent, and visualisations of attribution scores uses modified versions of the PyMOL API [138].

The PointVS code base is more than 12,000 lines of Python code in 60 files. It has the following functions:

[1]Makes use of Apple's 'Metal Performance Shaders' accelerated deep learning framework

- Training and testing of EGNN-based models for labelled protein-ligand pose classification and affinity regression, with a large number of hyperparameters and architectural choices available. Class system ensures prototyping of new architectures is straightforward.
- Automated conversion of molecular data from molecular data formats into a format appropriate for machine learning using filename pattern matching.
- Multiple methods for input attribution along with visualisation of attribution results in PyMOL.
- Logging of useful data both to disk and to the Wandb [139] online lab book.
- Unit tests and GitHub Workflows to ensure that updates do not break existing features.
- Various other scripts, written either as part of work that went into Chapter 4 or for other projects, a brief summary of which is included as Appendix D.

I will discuss some of these features in more detail here.

3.1 Model Class System, Training, and Validation

The first step in the development of PointVS was the selection of the model type. Although ultimately the GNN layer proposed by Satorras *et. al* [90] was used as the basis for the models in Chapter 4, two other candidates were tested. These used LieConv [85] and LieTransformer [87], both of which are discussed in Chapter 1.

Training and testing a new model architecture involves boilerplate code, such as iterating through a dataset, preprocessing inputs, logging results, and saving and loading models. This is required no matter the model architecture, and is time-consuming to write and test. Further, the models being tested as part of PointVS follow the same general pattern: first, input data is turned into a graph or point cloud which are then processed by the group-invariant layers, before pooling and a final fully connected layer. In order that prototyping new models is fast, a class hierarchy was established such that the functionality common to all models is abstracted away from any future developer. Only two methods must be written for a new architecture (as illustrated in Fig. 3.1).

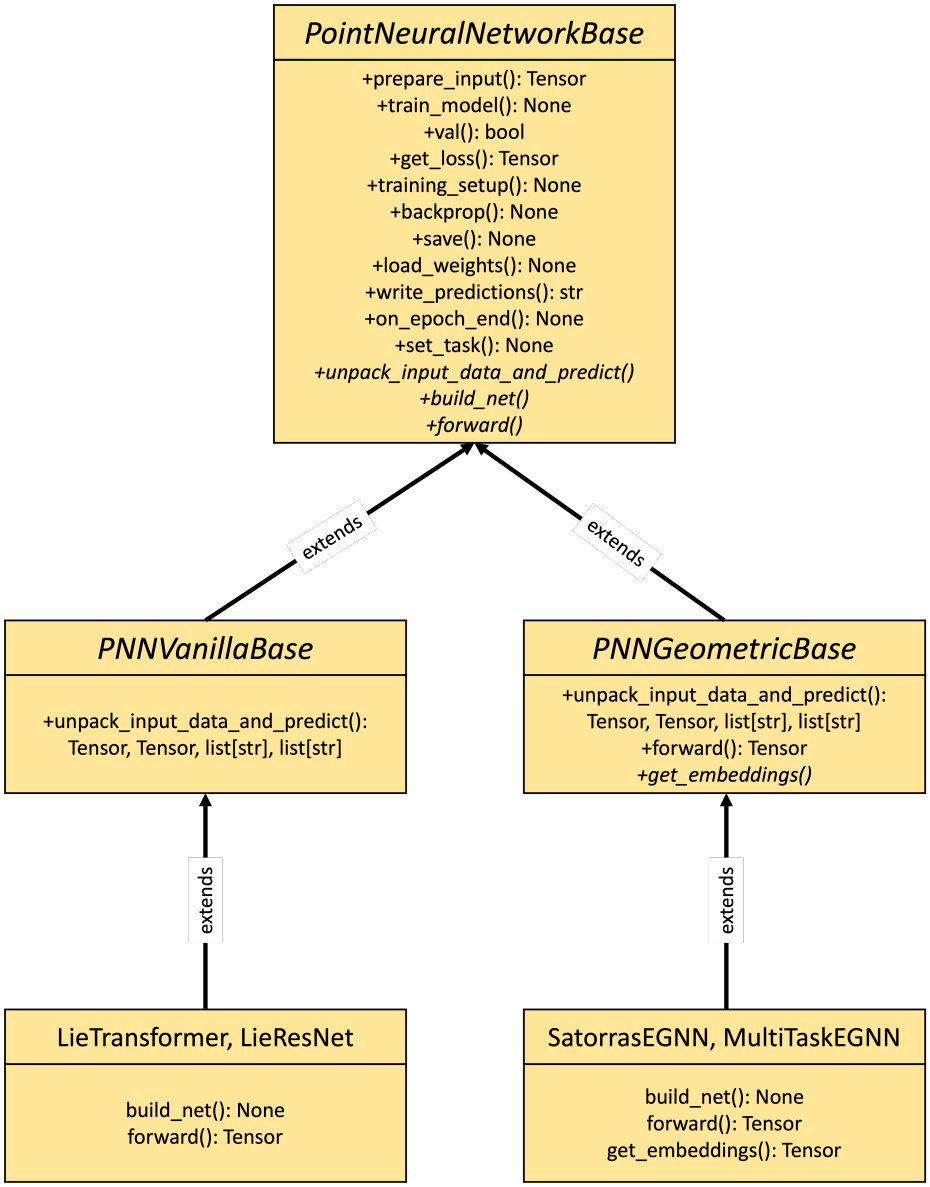


Figure 3.1: Class system for PointVS models. Abstract classes and methods are shown in itallics; all trainable (non-abstract) models must inherit either from PNNVanillaBase or PNNGeometricBase, and only two (three) methods have to be implemented for each new PNNVanillaBase (PNNGeometricBase)-based architecture. In the interests of brevity, variables are omitted from the diagram.

The abstract classes are:

- `PointNeuralNetworkBase`: Contains the logic for initialisation, training, validation, saving and loading weights, and logging
- `PNNVanillaBase`: Inherits from `PointNeuralNetworkBase`, expects input of standard PyTorch zero-padded graph tensor batches.

- `PNNGeometricBase`: Inherits from `PointNeuralNetworkBase`, expects input of PyTorch Geometric graph batches.

Trainable models must inherit from either `PNNVanillaBase` or `PNNGeometricBase`, with the former considered 'legacy', for use when PyTorch Geometric cannot be installed for some reason. The model used for Chapter 4 was `SatorrasEGNN`, which inherits from `PNNGeometricBase`. Prototyping new models is usually as simple as subclassing `PNNGeometricBase` and implementing the `build_net` and `get_embeddings` abstract methods. The first of these is where the network architecture is defined, and the second is how the inputs are processed to get individual node embeddings, which can then be pooled and processed into predictions. A subclass of `SatorrasEGNN`, `MultitaskSatorrasEGNN`, has also been provided which can switch between pose and affinity prediction.

The main driver script for training is `point_vs.py`. Eighty options for model training are made available as command line flags, which control hyperparameters, training and validation, and logging. All options are stored to disk, and there is an option to use Weights & Biases [139] for advanced logging. Inference can be carried out using `inference.py`, and interrupted training can be resumed using `resume_training.py`.

3.2 Visualisation of Input Attribution in PyMOL using PLIP

As mentioned in Chapters 2 and 4, one use for input attribution is as validation that models learn the rules of protein-ligand binding. However, another use case is for biochemists to be able to identify which interactions are important to the pose or affinity of a ligand in a binding pocket. A table of protein-ligand pairs and associated attribution scores could be used here, but a three dimensional image of individual atoms and the scores between them is more useful. Some examples can be seen in Chapter 4.

PyMOL [138] is a widely used molecular visualisation package, with free open-source options available on the Conda package management system [140]. It comes with a Python API, allowing users to programmatically interact with the visualisations in an intuitive, object-oriented fashion. This is leveraged by the Protein Ligand Interaction Profiler (PLIP) [141], which estimates non-covalent interactions using simple geometric rules, and produces PyMOL instances showing the output, for example

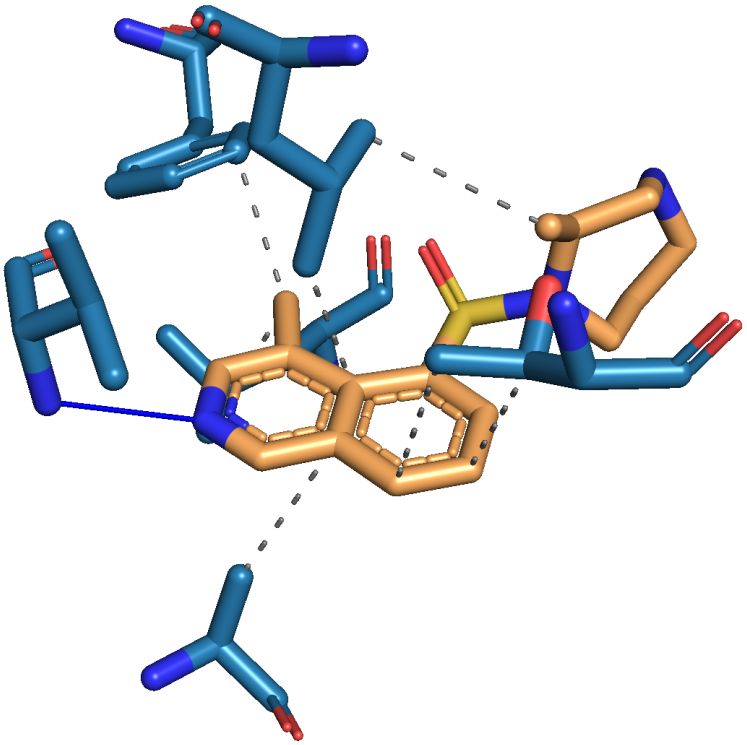


Figure 3.2: Example of the output of the PLIP package when run against entry 1Q8U of the PDB with default parameters. Blue lines are estimated to be hydrogen bonding interactions, with grey dashed lines estimated to be hydrophobic interactions.

in Fig. 3.2.

For the purposes of showing attribution scores in a similar way, some of the PLIP classes were inherited from and modified such that instead of showing the output of PLIP, they would show the output of the neural network. There is no reliable way to convert between the inconsistently labelled atoms in structures from the PDB and per-residue atom accounting system which the PyMOL API uses, so a system of coordinate matching had to be implemented using a `PositionDict` class. This can be conceived of as a 'soft lookup' for floating point numbers, which is robust to the problems caused by finite machine precision. In this way, atoms in the input, identified using the PDB standard, can be matched with their PyMOL equivalents. Once this is achieved, standard PyMOL API commands can be used to draw and shade bonds and atoms by their relative importance. The logic for all of this can be found in `plip_subclasses.py`.

3.3 Dataset Generation

The standard formats for storing information about small molecules are SDF and Mol2, and the usual format for macromolecules (such as proteins) is PDB. These contain a lot of information which is surplus to our requirements, and are also not suitable as raw input into neural networks. Parquet is an open source file format designed by Apache to be space efficient and fast to retrieve [142]. Conversion between large datasets in standard molecular formats and the parquets files that PointVS DataLoaders expect is therefore required. The two relevant scripts, `generate_types_file.py` and `types_to_parquets.py`, were also built as part of PointVS.

The `types` file format tells the PointVS `DataLoader` the paths of input parquet files as well as labels. This was is the format used by the authors of Gnina [24, 111]. The process for converting from SDF [143]/Mol2 [143]/PDB [144] to parquets files, then, is also mediated by a `types` file. The process is shown in Fig. 3.3.

The dataset generation process is split into two parts. Firstly, the directories containing protein and ligand structures in their original molecular file formats are searched, and a `types` file is generated (`generate_types_file.py`). This is used as input into another script (`types_to_parquets.py`), which converts the original structures into parquets format in such a way that the output directory structure is matches the types file. The new parquet files, along with the generated types file, constitute the dataset.

3.3.1 Types File Generation

In order to use a dataset, a PointVS `DataLoader` expects a text file containing labels and paths to parquet files containing ligand and protein structures (a `types` file). `generate_types_file.py` can be used to generate this from the original raw molecular data. `types` files for pose classification contain the following fields on each line:

```
<label> <docking_score> <rmsd> <receptor_path> <ligand_path>
```

The `label` is either 0 or 1, and the `docking_score` and `rmsd` fields can be optionally negative, indicating that they are unused. There are two options for generating the labels for pose classification datasets:

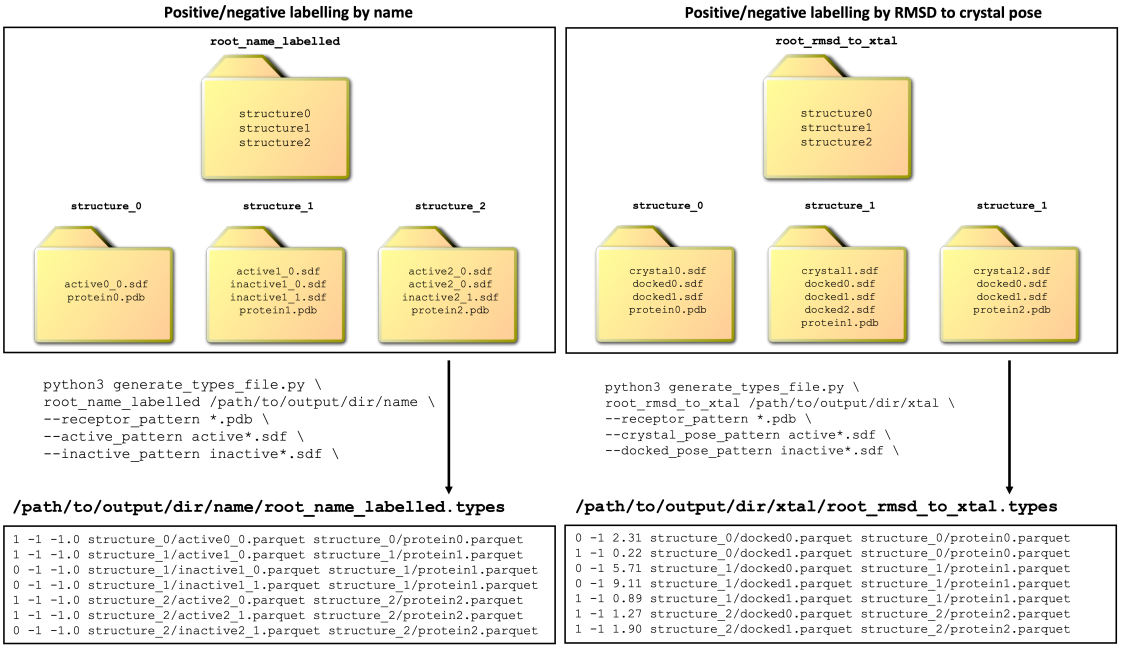


Figure 3.3: Generation of the types file. The Regex patterns supplied to generate_types_file.py dictate which files count as actives and decoys (left), or docked and crystal poses (right).

- **By name.** The ligand is labelled according to some naming pattern. In Fig. 3.3, 'correct' (1-labelled) poses follow the pattern `active*.sdf`, and incorrect (0-labelled) poses match `inactive*.sdf`.

- **By RMSD to crystal pose.** A ground truth crystal pose is provided, which should match the regex pattern supplied as `--crystal_pose_pattern`. The rigid RMSD distance between each of the molecules matching `--docked_pose_pattern` and the crystal pose is then assessed, and molecules with a larger than 2 Å are given a classification of 0 (1 otherwise). The third column of the output for this method is also populated with the RMSD values.

Types file lines for affinity prediction follow the format:

```
<pki> <pkd> <ic50> <receptor_path> <ligand_path>
```

As well as the `--crystal_pose_pattern` argument, a CSV file containing a PDB ID, the affinity, and the type (pK_{50}, pK_i or IC_{50}) of data must be supplied. The `--affinity` argument tells the program that the types file is to be used for regression. The two types of affinity data which are not supplied are set to -1.

3.3.2 Molecular File Formats to Parquet

As previously mentioned, the usual molecular file formats are not appropriate for fast processing into PyTorch graphs on the fly, so they must be converted to parquet format. Once a `types` file has been generated, it can be used along with the original SDF/Mol2/PDB files by `types_to_parquet.py` to convert the structures to the required format. OpenBabel [145] is used to read the molecular input files, as well as assigning atom types from their molecular environments. The directory structure of the output follows that laid out by the types file, in which all paths are relative to the root directory in which parquet files are saved. The full dataset is then comprised of the types file along with the parquet files.

3.4 Logging

It is important to be able to look at program outputs during training and testing, especially when developing new models. It is also important to log certain values, such as loss, mean prediction value on good/bad poses and validation performance. To this end, all program outputs are logged to file and `stderr`. This uses the standard library `logging.Logger` object, modified to better handle the logging of Pandas DataFrames. Machine learning outputs (loss, validation scores, etc.) are logged and uploaded using the `wandb` library to the Weights & Biases electronic lab book, where they are saved and can be displayed in various formats. All configuration options and hyperparameters are recorded, as well as the version of the code used, ensuring that experiments can be replicated.

3.5 Unit Testing and GitHub Workflows

PointVS has grown in scope beyond what was first intended, and so as with all software engineering projects, changing code in one place can cause unintended consequences elsewhere. Installation is also liable to break, given the large number of dependencies. To mitigate against these problems, unit tests were written with the Pytest framework [146] to cover important parts of the code base. These were added to GitHub Workflows, so that each time a commit is pushed to the remote repository, installation is checked and unit tests are run. The tests include:

- Model class instantiation (also tests whether most packages were installed correctly).

- Model consistency (model output should be deterministic function of input).
- Invariance of model output with respect to SE(3) transformations of input.
- Functions from `preprocessing.py` which act on input data to generate feature vectors/graphs.
- Checks on a script written for another student (`for_steph.py`, see below).

The GitHub Workflow also tests the import section of all Python files, ensuring that as well as being installable, the packages are in a usable condition. GitHub Workflows are in place for Linux and MacOS x86_64 setups; no GitHub-hosted Apple Silicon (AArch64) Workflow machines are available at time of writing.

3.6 One Use Case of PointVS: Exploring The Ability Of Machine Learning-Based Virtual Screening Models To Identify The Functional Groups Responsible For Binding

As described in the first two chapters, as well as in other publications [73, 74, 75, 147], the extent to which ML-based scoring functions are able to capture the physical interactions which govern binding energy and pose is uncertain. Experiments indicate that deep neural networks are prone to learn to make predictions based on ligand properties alone, and great care must be given to dataset construction and training to mitigate against this [63], and compelling statistical evidence that ML-based scoring functions learn a useful image of binding physics remains illusive.

There are two major obstacles to solving this issue. The first is that, as mentioned in Chapter 2, attributing predictions to certain parts of the input is difficult for deep learning models. The second is that, even if input attribution were solved, there exists no large dataset with the required ground truth against which to benchmark input attribution scores. As outlined in Chapter 4, the indication is that our particular GNN, PointVS, is able to extract important atom-atom interactions, but only on case studies and verified by visual inspection. A true ground truth validation set is required for an analysis with any statistical power.

With this in mind, Hadfield *et al.* have approached this second problem of ground truth from another angle. Instead of attempting to ascribe importance to non-covalent

interactions in real protein-ligand complexes, synthetic 3D protein-ligand complexes are generated along with contrived rules for what constitutes an active from inactive 'molecule'. Various ML-based scoring functions are then trained and tested against generated datasets, and their attribution scores are compared to the ground truth. Bias and noise in the training sets are carefully tuned, exposing the strengths and weaknesses of each scoring function when presented with common dataset flaws.

The main author of this work was Thomas E. Hadfield. My own role in this work was as follows:

1. Initial discussions and direction.
2. Code for synthetic dataset generation as modified SDF files.
3. Code for synthetic dataset integration with PointVS (see Chapter 4).

In the paper [105], a framework is proposed for assessing the ability of ML-based scoring functions to correctly assign importance to input atoms.[2] It uses PointVS GNNs, as well as random forests, to probe how robust input attribution is to ligand-specific bias in training sets, using examples from DUD-E and LIT-PCBA [76] as examples of biased training sets. The main results are that PointVS performs better both in terms of accuracy and input attribution in most realistic biased training scenarios; however, biased training significantly impairs the ability of the GNNs to accurately assign importance to input atoms. A preprint is available at [105].

The code described in this chapter was built primarily for a publication [106], which is described in the next chapter.

[2]The input atoms are comprised of a 'real' ligand (a molecule which could be observed in nature) and 'fake' protein 'pharmacophores'. In this contrived chemical universe, (unlike in real chemistry), like bonds with like - that is, hydrogen bond donor (acceptor) pharmacophores contribute to binding when they are close to hydrogen bond donors (acceptors) on the ligand. This quirk of nomenclature does not affect the experiments, however, as the only requirement for the contrived rules is internal consistency.

Chapter 4

A Small Step Toward Generalizability: Training a Machine Learning Scoring Function for Structure-Based Virtual Screening

The advent of GPU-accelerated deep learning and architectures such as CNNs still leaves the assessment of protein-ligand complexes to an accuracy comparable with lab-based methods an unsolved problem. This chapter seeks to build upon the work in the last, while addressing some of its shortcomings.

The discrete representation of space required by CNNs is not a natural way to represent atoms in 3D space. Discretisation introduces one of two problems. In the case of representing an atom's coordinates as those of its closest voxel as in [64], their positions are distorted. In the case of Gnina, the true atomic positions in 3D are encoded implicitly by the pseudo-density [111], at the cost of enormous redundancy: the standard representation of a 24 $Å^3$ binding pocket with 34 channels at a resolution of 0.5 Å requires 3.76 million floats. If there are 300 atoms in the binding site, this is the equivalent of more than 10^5 floats per atom.

GNNs solve these problems in featurisation. Atom types can be encoded as a small one-hot vector, and just three floats per atom are needed for their exact positions. In order to use these methods efficiently for protein-ligand interactions, invariance to $SE(3)$ transformations (all roto-translations in three dimensions) is required. Efforts over the last three years have provided myriad options to solve this problem [148]. It is one of these architectures in particular, the EGNN [90], that is used in this chapter.

The conflation, as described in the previous chapter, of pose validity with an arbitrarily defined binary 'would bind' vs 'would not bind' is easily solved. We can disentangle the training task into separate but related subtasks, both with distinct but important applications to SBVS:

1. **Post-docking pose classification.** Given an ensemble of possible docked pose for a ligand which we know or we presume bind in a particular pocket, classify them into similar or dissimilar to the pose observed by X-ray crystallography.

2. **Binding affinity prediction.** Given the structure of a ligand bound to a protein, assume that this is the correct pose and predict the strength of the interaction.

Given that which pose is adopted is a function of the overall change in energy of the bound state with respect to the unbound states, these two tasks are clearly related, which is something we can exploit. It is also easier to design training sets for the first task than it is to define actives and decoys as in Chapter 2. There are far more solved structures of protein-ligand complexes available than there are solved structures with measured binding affinity, and ligands can be redocked multiple times to give multiple examples of either class (a correct/incorrect pose). The ground truth as to what constitutes the 'correct' pose is already known and can be used to classify training data into (close to) correct and incorrect poses.

The objective of a ML-based scoring function is to approximate the distribution which takes two molecules as input and outputs the energy of their interaction. This distribution is dependent on interactions between the atoms of the two molecules and the solvent, and only a scoring function that accounts for these interactions can accurately predict binding affinity on unseen molecules. We show that a commonly used benchmark, CASF-16, overestimates the true accuracy of ML-based scoring functions when trained using the most commonly used training set. Ranking algorithms using this benchmark rewards memorisation of training data rather than knowledge of the rules of intermolecular binding. We present PointVS, a group-invariant graph neural network-based scoring function, which achieves state-of-the-art performance when tested on CASF-16 even after performing rigorous filtering of the training set. We then use attribution to demonstrate that PointVS is able to identify important interactions, and further, that it can be used to extract important binding pharmacophores from a given protein target when supplied with a number of bound structures. This

information is then used to perform fragment elaboration, and which shows improvements in docking scores compared to using structural information from a traditional data-based approach. This not only provides definitive proof that PointVS is learning to identify important binding interactions, but also constitutes the first deep learning-based method for extracting structural information from a target for molecule design.

This chapter is based on a publication [106]. First authorship is shared with Lucy Vost, who used my results as part of her extension of the STRIFE fragment elaboration algorithm [107]. The parts of the paper written in whole or in part by Lucy will be clearly indicated in section titles, and include all methodology, results and discussion pertaining to fragment elaboration. The only exception to this is the code and model training for extraction of hotspots from fragment screens.

The code developed for use in this project was used in other papers [105]. This use case, as well as some implementation details, is discussed in Chapter 3.

4.1 Introduction

As described in Chapter 1, drug design aims to optimise molecules to achieve a desired biological response. Typically led by human experts, this process is time-consuming and expensive. Computational techniques offer a promising alternate route to human-led design.

A key step in understanding a drug's behaviour is to understand how it binds to its target protein. Docking algorithms take as input the coordinates of protein and ligand atoms, and predict the possible conformations of the protein-ligand complex. There exists a function describing the energy of the complex, and as conformational space is searched, this function is minimised. In the case of a deterministic algorithm with a single starting conformation, a single output pose is generated along with an estimation of the binding affinity; for a probabilistic search, or any search with multiple starting conformations, the result is a list of binding poses. These are ranked in the order of estimated binding affinity. Affinities are calculated using a scoring function, which uses a combination of atomic interactions to approximate the binding energy. See Chapter 1, Section 1.2.2 for more details.

Over the last decade, machine learning models have shown promise in predicting both pose and energy in protein-ligand binding [149, 150], and in particular, deep learning models. The binding affinity of a given molecule to a protein target is a function of the positions and identities of its atoms, which can be approximated by a neural network in a ML-based scoring function. Given the coordinates of a protein-ligand complex (obtained by docking or otherwise), such a function considers all interactions between the two molecules. In this way, the interaction energies between two unseen molecules could be predicted. See Chapter 1, Section 1.2.3 for more details.

As outlined in Chapter 1, Section 1.2.3.3, recent research into machine learning-based scoring functions has highlighted their tendency to learn dataset biases, rather than physical interactions [73, 75, 74]. In these cases, the MLBSFs perform less of an assessment of atomic contributions, and something more analogous to a nearest-neighbours search in ligand-pocket space. This is useful if the application is to proteins or scaffolds with a large amount of previous binding data available, but not for new

targets or ligands, as the true functional relationships governing binding is not learned.

This problem is often exacerbated by a lack of rigorous dataset filtering. One of the most widely used benchmarks for scoring functions, both classical and machine learning-based, is the Comparative Assessment of Scoring Functions, 2016 (CASF-2016) [48]. The usual method of training a MLBSF is to train on the PDBBind [65] general set of crystal structures and binding affinities after removing the structures in CASF-2016. As we show in this paper, this approach suffers from information leakage concerning almost every test structure, resulting in an overestimation of the accuracy of MLBSFs.

There is an active area of research surrounding how machine learning models make predictions [151, 152], a technique known as input attribution. Such techniques can be applied to MLBSFs; the idea being that an ideal MLBSF—one which has learnt to distinguish atomic interactions, rather than just dataset biases—should be able to identify the most important binding interactions for a given protein target.

This binding information could then be leveraged in other stages of the drug development pipeline. In practice, attribution analyses depend on the architecture of the MLBSF. The two prevalent deep learning architectures used as scoring functions are convolutional neural networks (CNNs) [111, 153, 154] and graph neural networks (GNNs) [155, 156]. Attribution for CNN-MLBSFs is limited, because spatial relationships are lost as information flows deeper into the network [103]. GNNs, on the other hand, preserve the concept of the atom until the final layers of the network, so atom embeddings and edges can be interrogated for information.

Input attribution on MLBSFs is a promising approach for extracting binding insights from a protein (see Chatper 1, Section 1.2.3.8). For example, fragment-based drug discovery (FBDD) is a strategy that seeks to identify simple, small molecules (fragments) which interact with protein targets, and grow them into active leads (see Chapter 1, Section 1.1). In order to meaningfully add atoms to these molecules, structural knowledge about the target needs to be considered. This can be extracted from known binders, but this limits the exploration of chemical space as well as restricting the applicability of the approach to targets with known ligands. Extracting structural information directly from the protein itself is therefore advantageous. Currently, there is only one available generative model for fragment elaboration that relies

solely on protein structure [107]. This method uses a data-driven approach [108] to identify 'hotspots': regions in the protein pocket that contribute disproportionately to binding. Currently there exists no deep learning-based tool to identify such regions.

In this chapter we describe PointVS, a group-invariant GNN. It is a MLBSF designed to predict both binding affinity and pose score. We will show that it achieves competitive performance on the CASF-2016 benchmark—even after rigorously filtering the training dataset to avoid training on proteins or ligands similar to test structures—hence demonstrating its ability to learn spatial information. We then use attribution to show that the model successfully identifies important binding interactions, and further, that this information can be leveraged to perform fragment elaboration resulting in improved docking scores compared to using a data-driven approach. We make our code as well as our unbiased test and train splits available on GitHub at github.com/oxpig/PointVS.

4.2 Methods

An overall schema of the methods used to debias and test PointVS is shown in Fig 4.1.

4.2.1 Model

A graph is a natural way of representing a molecule, and attribution to individual inputs—namely atoms—is an intuitive process when using graph neural networks (GNNs). We use an architecture based on *E(n)*-Equivariant Graph Neural Network (EGNN) layers [90]. The $E(n)$ symmetry group contains all elements from $SE(n)$, as well as the group of all reflections; EGNN layers are also permutation-equivariant, meaning that the network is invariant to the order of the input vertices.

PointVS is a lightweight $E(n)$-equivariant graph neural network layer model, consisting of an initial projection to take the number of node features from 12 to 32, then 48 EGNN layers, followed by a global average pooling of the final node embeddings. This is followed by a sigmoid layer which gives a label $y \in [0, 1]$ during the first stage of training on pose prediction. During finetuning on affinity data, this final layer is replaced by an randomly initiated fully connected layer and ReLU activation, which outputs $\mathbf{y} \in (\mathbb{R}^+)^3$ (see Fig 4.2). This architecture includes residual connections for node features to allow for easier gradient flow and a richer combination of early and

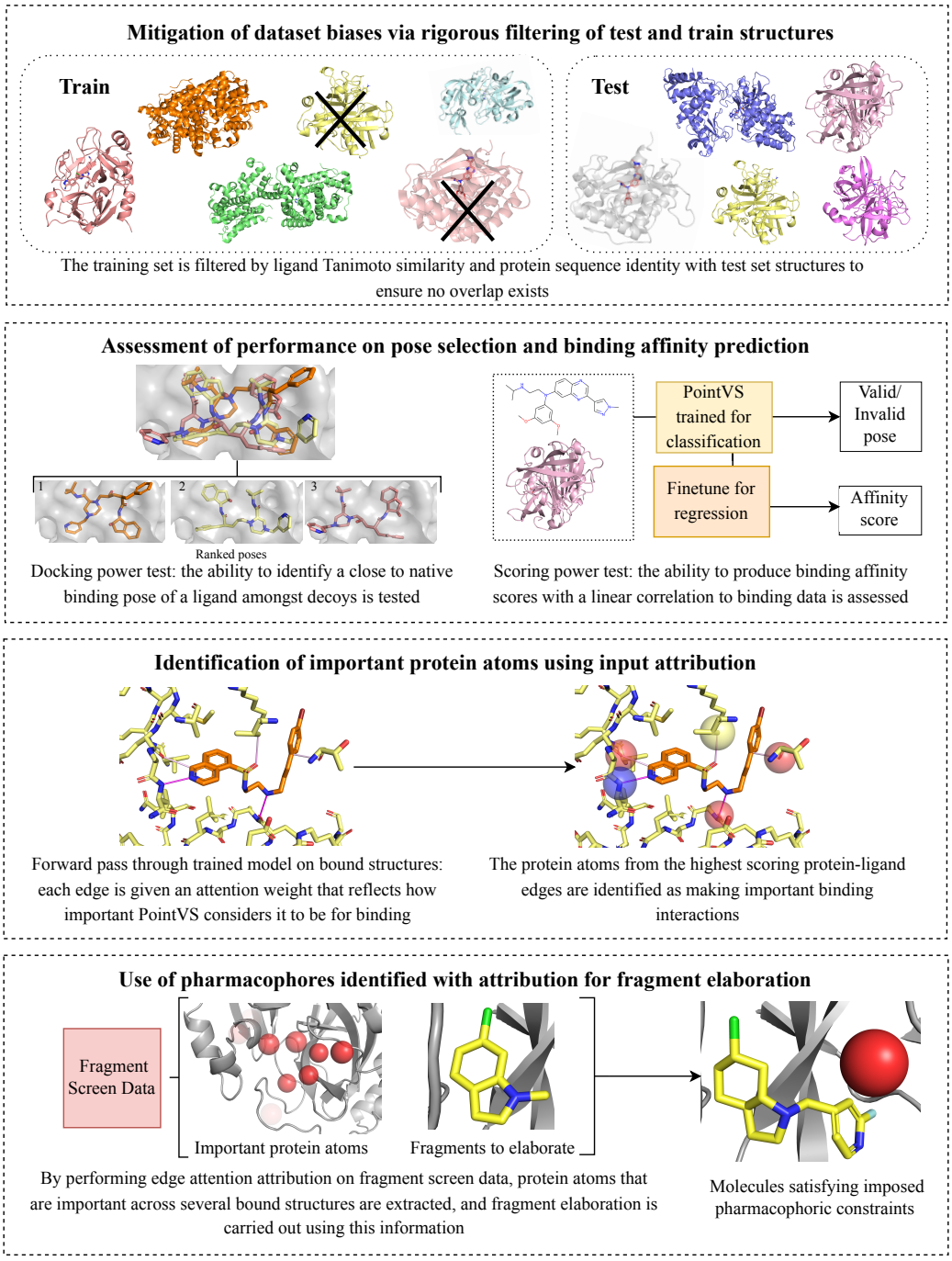


Figure 4.1: **Figure by Lucy Vost.** An overall schema of the methods used to debias and test PointVS. We first thoroughly filter the test and train sets, before benchmarking the performance of PointVS on the docking power and scoring power tests. We then use attribution to gain insights into important binding regions in the protein pocket, which we use for fragment elaboration.

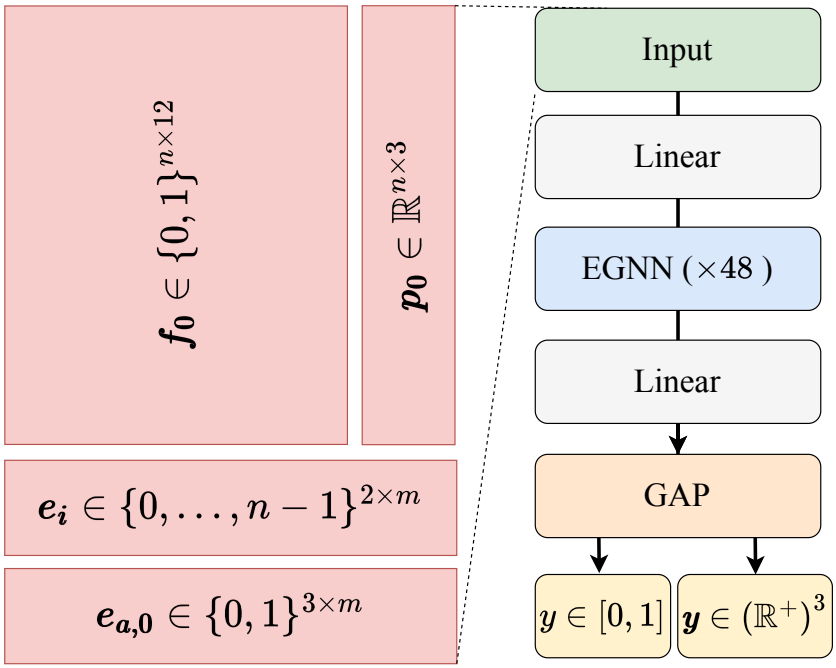


Figure 4.2: Architecture and input format of PointVS. Each of the n atoms in the input to the model (left) is given a one-hot encoded feature vector with a single bit added to indicate whether the atom is from the ligand or the receptor, as well as a position $\boldsymbol{p_0} \in \mathbb{R}^3$. There are m edges, defined by the edge indices $\boldsymbol{e_i} \in \{0, ..., n-1\}^{2\times m}$ which are the indices of connected atoms, and the corresponding edge attributes $\boldsymbol{e_a} \in \{0,1\}^{3\times m}$. These are one-hot encodings representing ligand-ligand, ligand-protein, and protein-protein edges. There are skip connections between each of the EGNN layers, and the linear and global average final pooling (GAP) layers only act upon the node features f.

late representations. It also uses a shallow neural network as an attention mechanism [157], which learns to score network edges—in this case, representing atomic interactions—by their importance.

4.2.1.1 Input Features

Our EGNN takes four inputs: positions, node embeddings, edge indices, and edge embeddings. If there are n atoms in an input structure, the positions tensor is an $n\times 3$ tensor containing the x, y and z coordinates of each atom. The node embeddings are an $n \times 12$ tensor of one-hot encoded OpenBabel atom types, with a bit to distinguish between ligand and receptor atoms. A summary of the five input matrices is shown on the left-hand side of Fig 4.2.

Edges are generated on-the-fly using a variable distance cutoff. We use a cutoff of 10 Å for ligand-protein edges, and 2 Å for ligand-ligand and protein-protein edges. In

this way, intramolecular edges closely mimic the covalent structure of input molecules, with a much more expansive connectivity to describe intermolecular interactions. The edge tensor is an $3 \times m$ matrix of m edges, with one-hot encoding for ligand-ligand, ligand-receptor, and receptor-receptor interactions.

All ligand atoms, as well as any protein atoms within 6 Å of any ligand atom is included in the input representation; the rest of the protein is ignored. This protein atom inclusion threshold is not the same as the cut-off used for constructing the graph edges from the node positions; because this threshold (6 Å) is lower than the protein-ligand edge cut-off (10 Å), any protein atoms that are included in the input representation are certain to be connected to at least one ligand atom by an edge. This ensures that there are no disconnected graphs made up solely of protein atoms.

4.2.1.2 Training

All models were trained for 10 epochs with a batch size of 32, using the Adam optimiser and cosine warm restarts with a maximum learning rate of 8×10^{-4} and a weight decay of 10^{-4}. For classification models, the selected model is the one with the best Top-1 value on the test set; for regression models, the selected model is the one with the highest Pearson's Correlation Coefficient (PCC) for affinity on the PDBBind Core Set.

For regression models with pretraining, once the classification task is complete, the best model is selected and the final linear layer with sigmoid is replaced by a multi-target regression, where the target is either pK_d, pK_i or pIC_{50}. The labels for the two types of affinity data not present in the training set are ignored for calculating loss, so those weights are unaffected.

4.2.2 Datasets

Our training dataset for pose classification is Redocked2020 [158] (Redocked from hereonin), with the test set generated according to McNutt *et al.* [111] so that classification performance can be directly compared with their work. This test is similar to the *docking power* test from CASF-16 [48], which we also conduct.

For regression on binding affinity values, we train a pose classifier on the Redocked set, finetune on affinity data from the PDBBind General Set, and test on the PDBBind Core Set. This replicates the *scoring power* and *ranking power* tests from CASF-16. As the Core Set is a subset of the General Set, we remove Core Set structures from the training data.

4.2.2.1 The Redocked Set

Training for pose classification uses the same training data as the Redocked model outlined by McNutt *et al.* [158]. In previous work by the same authors [111], ligands from the Pocketome v17.12 [159] were redocked using AutoDock Vina into their cognate receptors, with an RMSD calculated between the redocked poses and the crystal pose. Sub-2 Å poses were labelled as active poses, with the rest labelled as inactive. Further negative examples were generated adversarially: a Gnina CNN trained on the original redocked dataset was used to optimise docked poses. Optimised poses which were scored highly (> 0.9) by the trained CNN but were more than 2 Å RMSD from the crystal pose were added as inactive poses. In total, there are 52,385 active and 734,417 inactive poses Redocked.

4.2.2.2 PDBBind Refined: GninaSet$_{Pose}$ Set

The PDBBind refined set v.2019 [65] is a carefully selected set of high-quality protein-ligand complexes, complete with binding affinities. In their Gnina 1.0 paper [111], the authors took the 4,852 unique protein-ligand complexes, removing solvents and any other molecules besides the protein and the ligand, as well as removing structures with ligands that do not have a mass between 150 and 1,000 Da. This filtered, preprocessed version of the refined set is hosted by the authors of Gnina 1.0 [111] at https://bits.csb.pitt.edu/files/gnina1.0_paper. Gnina was used with default settings, excepting `-num_modes=50`, `-min_rmsd_filter=0.0`, and the use of the standard Vina scoring function in refinement and ranking rather than a CNN.

Of the 4,260 filtered structures, the 441 which are not used in training the 'redocked' Gnina model are selected as the test set for consistency with that work, which we call GninaSet$_{Pose}$. Up to 50 poses are reranked by our GNN (some of which may be very similar), after which the standard 1.0 Å RMSD filter is applied to get a final pose ranking.

4.2.2.3 PDBBind General and Core Sets

The PDBBind v.2020 General set contains protein-ligand complexes and binding affinities curated from the PDB [68]. The affinities are in the form of pK_d, pK_i or pIC_{50}, and this was the training set for affinity regression. The PDBBind Core set, also known as the CASF-2016 test set for affinity regression, comprises 285 high-quality crystal structures, as well as their binding affinities [48], and is a subset of the General set. This is the test set for regression (*scoring power* and *ranking power*).

Dataset	Size	Unique PDB IDs	Task/ Split
Redocked	780,572	19,595	C/Train
Redocked\GninaSet$_{Pose80}$	633,047	17,528	C/Train
Redocked\GninaSet$_{PoseR}$	633,047	19,002	C/Train
Redocked\Core$_{80}$	629,963	16,900	C/Train
Redocked\Core$_{R}$	629,963	18,852	C/Train
General	19,157	19,157	R/Train
General\Core$_{80}$	14,555	14,555	R/Train
GninaSet$_{Pose}$	20,411	411	C/Test
Core	285	285	R/Test

Table 4.1: Datasets, their sizes and the number of unique PDB structures in their generation. Redocked includes no structures with the same PDB code as any structures in the GninaSet$_{Pose}$ or Core sets, and the General set shares no PDB codes with the Core set. In the Task/Split column, 'C' refers to pose classification and 'R' refers to affinity regression.

4.2.2.4 Training dataset filtering

Due to the lack of affinity data, we choose to pretrain on a pose classification task in order to learn some of the physics of binding before finetuning on affinity data. As per Fig 4.1, filtered subsets of both the Redocked set and the PDBBind v.2020 General Set are generated for both of these stages of training, as well as random subsets of the original training sets of the same size as the filtered sets. The filtered datasets were constructed by removing proteins and ligands if they met either of the following criteria:

1. Tanimoto similarity of the 2048-bit, 3-radius Morgan fingerprint between the ligand and any of the 285 test set ligands is greater than 0.8

2. Sequence identity between the protein and any of the 285 test set proteins is greater than 0.8

These filters were also applied to the Redocked set with respect to the GninaSet$_{Pose}$ test set. All files which could not be parsed by either RDKit [160] or OpenBabel [145] were also discarded.

This process reduces the training set sizes from 780,572 to 633,047 and 629,963 (Redocked\GninaSet$_{Pose80}$ and Redocked\Core$_{80}$) and from 19,157 to 14,555 (General\Core$_{80}$). A full list of datasets is given in Table 4.1. In order to deconvolute the effects of training set size from the effect of the filters, we also constructed training sets of

the same size, randomly sampled from the original training sets. These are called Redocked\GninaSet$_{PoseR}$, Redocked\Core$_R$ and General\Core$_R$. Carrying out filtering in this way should ensure the PDBBind Core set performance is a more accurate reflection of accuracy on unseen targets and ligands.

4.2.3 Attribution

It is well documented that the CNNs currently available for virtual screening are prone to distinguishing binders from non-binders based not on protein-ligand interactions, but on ligand features alone [161, 74, 75, 73, 147]. Ensuring that predictive power is diminished in the absence of receptor information in the test set causes at least some receptor information to be used [63]. However, only directly observing which parts of the input space are used in classification can give affirmative proof that machine learning models are learning to identify physical interactions that separate active from inactive poses.

Attribution in machine learning for pose prediction and virtual screening is an ongoing problem; the architecture of most CNNs causes the concept of individual atoms to be lost as information flows deeper into the network, making them fundamentally unsuitable to attribution at the atomic level. By contrast, graph-based architectures such as PointVS maintain distinct nodes (atoms) until the final layers. These, having passed through the entire network, are rich in information and are directly related to the relevant atom in the input. As the node features are directly attributable to their input atom, and the final pose score is a function of the node features, there is a direct relationship between the atoms and the PointVS score, although there is no incentive during training to localise class contributions on particular atoms. The edges between atoms can also be probed for how much importance is ascribed to atomic interactions, as an intuitive way to describe non-covalent bonds. As per Fig 4.1, we can also use this knowledge of importance assigned by PointVS to bonds to identify important protein atoms for binding.

We use two approaches for attribution: atom masking for Gnina, and edge attention for PointVS.

4.2.3.1 Atom masking with Gnina

Masking is a process by which the importance of different atoms can be ascertained. It is carried out by calculating the difference between the score given when an atom is removed from the set of input atoms, and the score when it is present. For a more thorough explanation, see Section 2.3.3 and Fig. 2.8.

4.2.3.2 Edge attention attribution with PointVS

Each layer of PointVS has an attention-like mechanism where the edges are given a score between 0 and 1 by a shallow multilayer perceptron (MLP), which takes as input the edge embeddings themselves. The edge message passed to the i^{th} node at each layer $\mathbf{m}_i$ is the sum of connecting edge embeddings $\mathbf{m}_{ij}$ in the neighbourhood of the node $N(i)$, weighted by this attention score e_{ij}, as in Equation 8 in Satorras *et. al* [90]:

$$\mathbf{m}_i = \sum_{j \in N(i)} e_{ij} \mathbf{m}_{ij}$$

The attention weights can be thought of as indicating how much each edge (atom-atom interaction) is weighted by the network. A schematic of how attention works for EGNN layers can be found in Chapter 1 (Fig. 1.5).

4.2.4 Fragment Elaboration (carried out by Lucy Vost)

To elaborate a fragment towards more potent, lead-like compounds, information about important areas of the protein pocket being targeted is key. We set up a series of tests to investigate if the attribution scores from PointVS can identify these important sites.

By performing attribution on a crystal structure of a given protein with a small molecule bound, we obtain binding information in the form of an importance score for every protein atom less than 6Å away from any ligand atom. If a fragment screen has been carried out on said target, the attribution process can then be repeated for several crystal structures to obtain an average importance score for each of the protein atoms. We use this list of protein atoms and associated scores as hotspots in fragment elaboration experiments. To assess them, we say that the quality of a hotspot— how important the point it is placed on actually is for binding—is proportional to the binding affinity of the molecules generated in the elaboration process, which we assess using a metric derived from docking score.

We compare the ability of PointVS hotspots to highlight important binding regions to traditional, data-based fragment hotspot maps. To obtain these, we use the Hotspots API [108], which implements the algorithm described in Radoux *et al.* [162]. We then process the output of the API as in Hadfield *et al.* [107].

For a given protein target, the API calculates acceptor, donor and apolar atomic propensity maps using SuperStar [163]. This tool defines a grid covering the protein with points spaced 0.5Å apart, and then using data from the Cambridge Struc-

tural Database (CSD) [164], assigns a propensity for the three atomic probe types—acceptor, donor, and apolar—to each point. These scores are assigned based on the principle that if a positive interaction between two groups at a certain distance and angle is made, then it will occur more frequently in structures stored in the CSD, and thus be assigned a higher propensity score. These grid point scores are then weighted according to how buried in the protein they are, so that more buried sites are favored.

Further scores are then calculated with small chemical probes consisting of aromatic rings with an acceptor, donor, or apolar atom in the substituent position. The three weighted propensity grids are sampled by their corresponding probe: a given probe is translated to all grid points with scores above a threshold of 15, and randomly rotated 3000 times around the centre of the substituent atom. For each pose, the probe atom scores are read from their corresponding grids, and the probe score is found as the geometric mean of the scores of the probe's atoms. Once complete, the sampled probe scores are assigned to an output grid.

To process the donor and acceptor grids, a greedy clustering algorithm is employed. A cluster is initialised as a random single point, and all points less than 1Å away that are not part of another cluster are added. Then, for each point added, this process is repeated, so all remaining unclustered points within 1 Å are added. This process continues until there are no unclustered points within 1 Å of the points forming the cluster. Once a cluster is finalised, a new cluster is defined by selecting another single unclustered point. This process is repeated until all points have been assigned to a cluster. The centroids of these clusters are defined as the mean position of the points constituting them. Clusters comprising less than eight points are removed, as well as those with centroids outside of an apolar region (this is assessed using the apolar grid). The remaining clusters are then allocated a score consisting of the number of points comprising them, and output alongside their associated scores as the acceptor and donor hotspot maps for a given protein target.

For each target we perform fragment elaboration tests on, we obtain hotspot maps with PointVS using the crystal structures of their fragment screens available on the Fragalysis platform [165]. The structures used are listed in Appendix A. To obtain traditional hotspot maps, we run the Hotspots API on one structure of the target randomly selected from the Fragalysis set of bound structures.

We also use the Fragalysis set of crystal ligand structures listed in Appendix A to obtain fragments to elaborate. To do this, we enumerate all cuts of acyclic single bonds that are not part of functional groups, and add a 'dummy' atom at the site

of the cut. This atom is where atoms will be added to the fragment; this process produces 109 fragments to elaborate from.

To perform fragment elaboration using the two sets of hotspot maps, we use STRIFE, a generative model for fragment elaboration. We follow the method outlined for its use in Hadfield *et al.* [107]. We take as input a fragment, a specified exit atom, and information about one hotspot; namely, its coordinates and type. STRIFE then generates molecules matching the pharmacophoric profile specified.

To achieve this, STRIFE has two main stages. The first of these, exploration, aims to generate a set of elaborations which contain an acceptor or donor in close proximity to a hotspot. These elaborations are then docked using GOLD's constrained docking functionality [34], and those which successfully place a functional group within a certain distance (2 Å for the API hotspots, which are in the protein pocket, and 3 Å for PointVS hotspots, which are on protein atoms) of the hotspot being targeted are selected as 'quasi-actives'. In the next stage, refinement, these quasi-actives are used to derive a fine-grained pharmacophoric profile. STRIFE then generates elaborations using those pharmacophoric profiles, and docks them. These docked elaborations constitute its final output.

For a given hotspot, we perform elaboration on a set of fragments. The size of this test set depends on the number of bound structures of ligands available for the target in question (see Appendix A). For some fragment-hotspot pairings, elaborations cannot be generated. This can be due to the hotspot in question being too close or too far from the exit point on the fragment, or at an angle that is unfavourable to elaborate along. In these cases, the generation stage will be unable to produce any quasi-actives; in other words, it will be unable to generate any molecules that, when docked, have a hydrogen bond acceptor or donor within a certain distance from the hotspot. An example of this is shown in Table 4.6, which shows the number of fragments successfully elaborated in the pocket of Mpro (SARS-CoV-2 Main protease) for every hotspot tested. We see that the hotspot ranked ninth by the Hotspots API, for instance, cannot be reached by elaborating on any of the 109 fragments tested, and hence no molecules are output.

The number of final elaborations generated is dependent on the number of quasi-actives that the first stage, exploration, generates. Although STRIFE is asked to generate 250 elaborations per fragment, there are often fragment-hotspot pairings where the number of quasi-actives identified is insufficient for the refinement phase to produce 250 distinct molecules. The mean number of elaborations generated for a given pair is then less than 250, as seen in Table 4.6. Nevertheless, as each hotspot

is tested on a number of fragments occupying various positions in the pocket, we still obtain a large number of generated molecules for a given hotspot.

To assess the generated molecules, we dock them using GOLD [34], the CCDC's protein-ligand docking software. We use the constrained docking functionality—constraining using the fragment, for which we have a crystal structure—from which each molecule receives a score. We then calculate a ligand efficiency by dividing this score by the number of heavy atoms present in the molecule. This is to compensate for docking algorithms tendencies to favour larger molecules. From the ligand efficiency, we derive a standardised ligand efficiency. This is obtained by standardising the ligand efficiencies of the generated molecules and the ground truth molecule for given fragment-hotspot combination to have zero mean and unit variance (here, ground truth molecule refers to the molecule that was segmented to obtain the test fragment). We then sort the standardised ligand efficiencies of the elaborations from highest to lowest scoring, and take the mean of the top α. In this work, we use α=20. Subtracting the ground truth ligand efficiency value from this value then provides us with ΔSLE_{20} for every fragment-hotspot pair, so for a given hotspot, we take the mean over the ΔSLE_{20}s of all the fragments successfully elaborated towards it. We use this metric as it provides an insight into how the elaboration process when using a particular hotspot has impacted the original molecule's docking score: a positive $\Delta\text{SLE}_{\alpha}$ means the top elaborations are improvements on the original molecule, and a negative $\Delta\text{SLE}_{\alpha}$ shows that the elaboration process has decreased the ligand efficiency with respect to the starting molecule.

4.3 Results

We test the ability of our machine learning-based scoring function, PointVS, to perform pose selection and affinity prediction. Through the use of attribution, we also verify that not only has PointVS successfully learnt to recognise binding interactions, but that when given a number of bound structures of a given target from a fragment screen, it can be used to extract information about important binding pharmacophores that can then be used in automated fragment elaboration experiments.

4.3.1 Bias in the CASF-16 Test Set

We implemented a rigorous filtering step to ensure that the training set and test set structures used by PointVS have no overlap (see Section 4.2.2.4). In order to compare PointVS with other machine learning methods which have not included this filtering

step [166, 167, 111], we also constructed a smaller test set out of the PDBBind Core set by applying the same filters but excluding test set structures rather than train set structures. However, we found that such a test set would only contain a single structure: 284 of 285 of the Core set proteins have a counterpart with 90% sequence similarity in the General set. Further, 273 of them (95.7%) have a counterpart with an identical sequence, making any fair and unbiased comparison to these methods impossible.

4.3.2 Docking Power

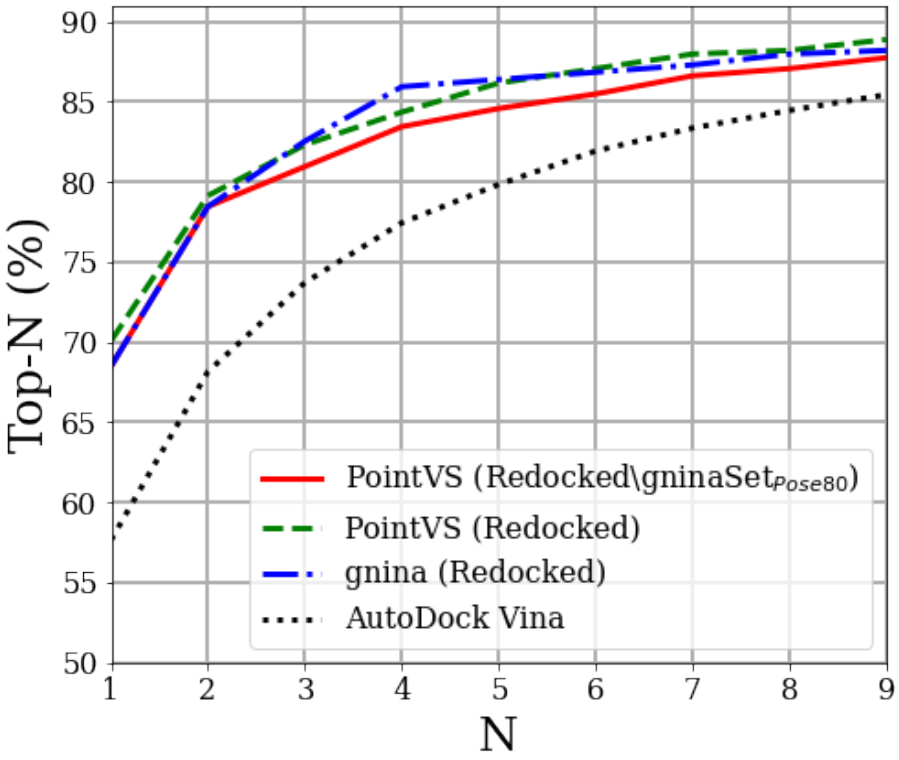


Figure 4.3: Top-N vs N pose ranking performance on the GninaSet$_{Pose}$ set for AutoDock Vina, PointVS and Gnina. Terms in brackets in the legend refer to the training sets; filtering the training set by protein and ligand similarity results in slightly degraded performance. The Top-1 values for PointVS trained on Redocked\GninaSetq$_{Pose80}$ and Gnina trained on Redocked are the same (68%), with PointVS trained on Redocked achieving 70%.

Docking power refers to a scoring function's ability to identify a ligand's native binding pose among decoys. It is quantified using Top-N, which is the percentage of systems with a "good" pose ranked in the top N. A pose is considered good if the RMSD to the crystal pose is less than 2Å.

Model	Top-1	
	Crystal pose included	Crystal pose not included
PointVS (Redocked\Core$_R$)	90.5	84.9
PointVS (Redocked\Core$_{80}$)	91.2	84.2
PointVS (Redocked)	**91.3**	**85.3**
Gnina (Redocked)	91.2	83.2
AutoDock Vina	90.2	84.6

Table 4.2: Top-1 pose ranking performance for different models and training sets (in brackets) on the PDBBind Core set (the CASF-16 docking power test set). The non-machine learning scoring function is shown in italic. The performance is shown for both the case where the crystal structure of the ligand is included in the set of poses being ranked, and the case where it is not included, in which case the native pose is defined as any pose less than 2Å RMSD away from the crystal pose. The highest Top-1 for each case is shown in bold.

Top-N pose ranking performance on the GninaSet$_{Pose}$ set is shown in Fig 4.3. When the training sets are the same, PointVS ranks a good pose at the top in 70% of cases, as opposed to 68% with Gnina. When PointVS is trained on the smaller Redocked\GninaSet$_{Pose80}$ set, such that no similar proteins or ligands as the test set are seen during training, this drops to 68%. This suggests that for pose prediction, filtering the dataset to remove overlap between train and test structures only causes a minor decrease in docking power.

The Top-1 pose ranking results for the Core set (the CASF-16 docking power set) are shown in Table 4.2, for both the case where the crystal pose is included amongst the poses to be ranked, and the case where it is not.

The difference in performance between all of the methods is minimal, and is not affected significantly by whether or not the training data has been filtered for structural similarity (the without-ligand test set scores are 85.3%, 84.9% and 84.2% for PointVS trained on Redocked, Redocked\Core$_R$ and Redocked\Core$_{80}$, respectively). This lack of dependence on debiasing is in agreement with the docking power results shown in Fig 4.3, although the performance of AutoDock Vina is comparable with the machine learning methods in this test.

4.3.3 Scoring Power

The scoring power is defined by Su *et al.* as "the ability of a scoring function to produce binding scores in a linear correlation with experimental binding data" [48].

This is best measured using the Pearson correlation coefficient (PCC) between the score given by the scoring function and the measured binding affinity data. For training our networks, we once again use both filtered and unfiltered versions of the PDBBind General dataset to to test the influence of dataset bias.

Scoring Function	PCC
PointVS (General)	**0.805 ± 0.010**
PointVS (General\Core$_R$)	0.803 ± 0.012
PointVS (General\Core$_{80}$)	0.754 ± 0.015
ΔVinaXGB	0.796
Gnina (Redocked)	0.753 ± 0.008
ΔVinaRF	0.732
X-Score	0.631
AutoDock Vina	0.601

Table 4.3: Pearson correlation coefficients between measured affinity and different scoring functions on the Core set. Training sets are shown in brackets. Non-machine learning scoring functions are shown in italic. The highest PCC is shown in bold.

We train PointVS to classify on Redocked\Core$_{80}$, and then switch out the final layer for multi-target regression trained on General\Core$_{80}$. We also use the Gnina model trained on the PDBBind General set supplied as part of work by McNutt *et al.* [111] to find the PCC on the Core set, but again emphasise that the training data for this model includes many proteins and ligands which are similar or identical to those in the Core set. This model achieves a PCC of 0.816. The results from other methods can be seen in Table 4.3. The three best methods without dataset filtering (Gnina, PointVS and ΔVinaXGB) achieve very similar performance, and PointVS outperforms the best non-machine-learning methods from the original CASF-16 work [48], AutoDock Vina, significantly (0.805 vs 0.601). However, the performance of PointVS when trained on the filtered dataset General\Core$_{80}$ is a better indicator of its true performance on unseen protein-ligand combinations with no analogues in the training data.

The three machine learning methods which use three distinct featurisation methods and three separate architectures (PointVS, Gnina and ΔVinaXGB) obtain similar PCCs when trained on the unfiltered General set. This suggests that the limiting factor in predictive performance is what information the training data holds about the test set data, rather than the architecture or featurisation.

To test whether the drop in the performance of PointVS when we remove similar structures from the training set is related to bias or change in training data size, we trained PointVS on General\Core$_R$. This has the same training set size as General\Core$_{80}$, but can still contain similar structures to those in the test set. PointVS trained on the General\Core$_R$ achieves a PCC of 0.803 on the CASF-16 set. This is almost identical to the PCC of PointVS trained on the entire General set (PCC of 0.805), and significantly higher than PointVS trained on the General\Core$_{80}$ set (PCC of 0.754). This once again points to the performance of these methods when trained and tested using biased datasets being boosted. This contrasts with our findings in pose prediction performance, in which we saw very minimal change when filtering was performed.

As PointVS trained and tested in a debiased way still appears to be predictive of binding affinity, we next explored whether it has learnt to identify important interactions in the binding site.

4.3.4 Attribution

PointVS appears to offer a substantial improvement in predictive performance of binding affinity over non-machine-learning methods. However, MLBSFs are notoriously difficult to interpret. Good performance when trained on debiased training data is circumstantial evidence that the interatomic interactions involved in binding are being learned, but demonstrating that the attribution scores of PointVS relate to important binding interactions would be more concrete. As discussed in Chapter 1, Section 1.2.3.8, GNNs are well-suited to this kind of analysis, and we next present several results demonstrating that important interatomic interactions have been learnt by PointVS.

4.3.4.1 Human Tankyrase-2 Inhibitors

The Protein-Ligand Interaction Profiler (PLIP) [141] uses simple geometric rules to identify interactions at the interface between a protein and a ligand. We use PLIP to identify non-covalent bonds for three human Tankyrase-2 inhibitors, and see if the atoms forming these bonds are highlighted when attribution is carried out on Gnina [111] and PointVS.

In their paper on attribution with Gnina [103], the authors describe various methods to overcome the difficulties associated with performing attribution on a CNN. However, the code implementing these methods, Gninavis, is currently unavailable,

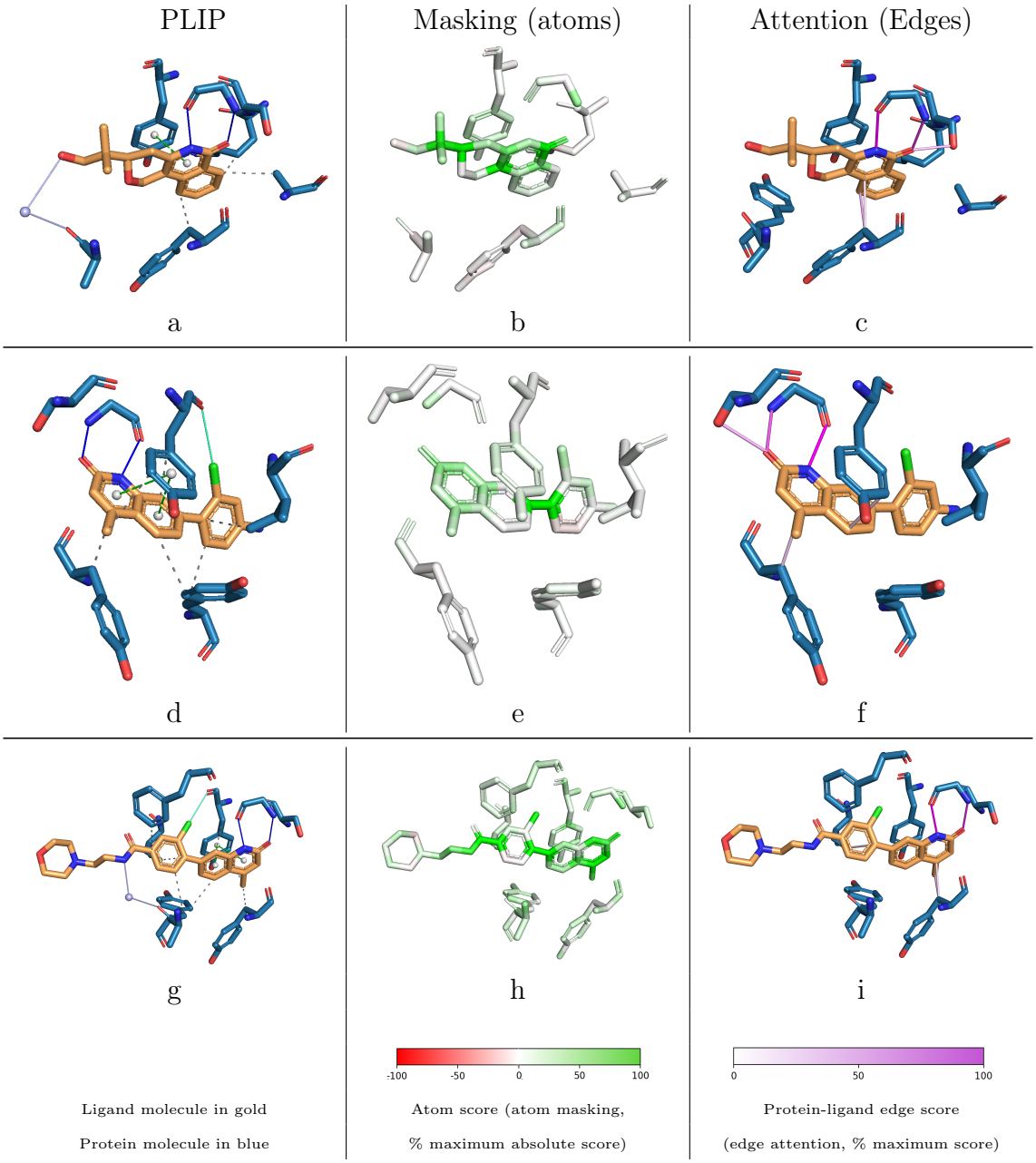


Figure 4.4: Three Tankyrase-2 inhibitors: 5C5P (a, b, c), 4J21 (d, e, f), and 4J22 (g, h, i). The leftmost column shows the Protein-Ligand Interaction Profiler (PLIP) analysis of each structure, where dark blue lines are hydrogen bonds, dotted grey are hydrophobic interactions, dotted green are π-π interactions, solid green are halogen bonds, and lilac are water bridges. The central column shows the results of performing masking with Gnina, with green representing a positive attribution score (identified as making a positive contribution to binding) and red a negative score (making a negative contribution). The rightmost column shows the results of edge-attention attribution performed with PointVS, with the lines between atoms showing the top five highest-scoring protein-ligand edges for each structure, with darker pink representing higher scores.

so we instead implemented a standard atomic masking procedure as described in Section 2.8.

The structures of three tankyrase inhibitors are shown in Fig 4.4. Tankyrase-2 was chosen as it is a common target for cancer therapeutics, with high-quality structures available in the PDB. The leftmost column (panels a, d and g) shows the PLIP analysis of each structure. The structures in the other two columns show the results of attribution with Gnina (centre, panels b, e and h) and PointVS (right, panels c, f and i). The structures in the central column are thus coloured by the attribution scores obtained from masking with Gnina, with atoms contributing negatively to binding being shown in red (negative score), neutral atoms in white, and positively-contributing atoms in green (positive score). Attribution with PointVS was carried out using edge attention, and thus the attribution scores produced correspond to interactions between pairs of atoms rather than individual atoms. The five highest-scoring edges are shown in panels c, f and i, with dark pink lines representing high scores, and lighter pink representing lower scores.

In Fig 4.4b the two ligand atoms that form hydrogen bonds are scored by Gnina as the 6th and 8th most important of 20. Of the two hydrogen bonding protein atoms, Gnina recognises one of them as being extremely important (the highest scoring protein atom), and the other as neutral. Fig 4.4e tells a similar story, with the three important ligand atoms scoring 4th, 7th and 8th out of 20, but the protein atoms they respectively bind to scoring negatively, neutrally and very slightly positively. In the third structure, Fig 4.4h, the three bound ligand atoms are scored 11th, 13th and 14th of 30 ligand atoms, with the corresponding protein atoms scoring negatively, weakly positively and moderately positively. Interestingly, the last two structures both contain halogen bonds, and in both cases Gnina ranks the ligand atom as important, and the protein atom as a negative contribution. Overall, Gnina struggles to identify important individual atoms (both in the ligand and protein) for binding. The atoms being scored individually rather than by pairs as with edge attention makes it more difficult to identify the bonds that are being formed: of the bonds identified by PLIP, only in the first structure was there an example where both the protein and ligand atoms were given a high score, making it the only case where an important bond was identified.

When we perform atom masking on PointVS, we see that the results are similar to those from Gnina, in that there is no discernible relationship between atoms involved in bonding according to PLIP and their PointVS masking scores. However, we find that edge attention analysis provides a clearer insight into important interactions.

In Fig 4.4c, we see that the bonds ranked first and second most important by PointVS correspond to the hydrogen bonds identified by PLIP in Fig 4.4a. Edges between the two aromatic rings involved in π-stacking are also given high importance scores, being ranked fourth and fifth most important. Similarly for Fig 4.4f, where we see the atoms involved in hydrogen bonds in Fig 4.4d being connected by edges assigned high importance by PointVS (ranked first and second most important). We also again see high scores assigned to the edges connecting the rings involved in π-stacking. PointVS does, however, fail to recognise the edge joining the atoms involved in the halogen bond shown in green in Fig 4.4d. PointVS also identifies the hydrogen bonds shown in Fig 4.4g as being important, again ranking them as the most and second most important edges in Fig 4.4i. The other edges assigned high importance correspond to hydrophobic interactions. These results suggest that the edge attribution scores from PointVS are able to identify important binding interactions.

4.3.4.2 Large Scale Attribution Tests (carried out by Lucy Vost)

To further verify that performing attribution with PointVS results in similar bonds being highlighted to those specified by PLIP, we perform attribution on the PDBBind Core set (the CASF-16 test set). We then calculate the PCC between the scores assigned to the top five and top ten scoring protein atoms, and the distance between the protein atom and the corresponding ligand atom. This results in a PCC of -0.766 for the top five protein atoms, and -0.844 for the top ten protein atoms. This strong negative correlation implies that protein-ligand atom pairs that are close together (which in turn PLIP will consider more likely to be forming a bond) are commonly identified by PointVS. We also calculate the correlation coefficient between the ranks of the top five and ten protein atoms and their PLIP rank (this corresponds to the protein atoms ranked by distance to the nearest ligand atom), and obtain 0.719 and 0.788, respectively.

We also carry this process out for Gnina, but as masking is a significantly more computationally challenging process than edge attention analysis, we randomly select 20 structures from the Core set to carry this out on. The PDB IDs of these structures are given in Appendix B. We also perform attribution with PointVS on this subset for comparison. The results of calculating both the PCC and the rank correlation, ρ, are shown in Table 4.4.

We again see a strong negative correlation between the score assigned by PointVS and the distance between the atoms, and a strong positive correlation between their

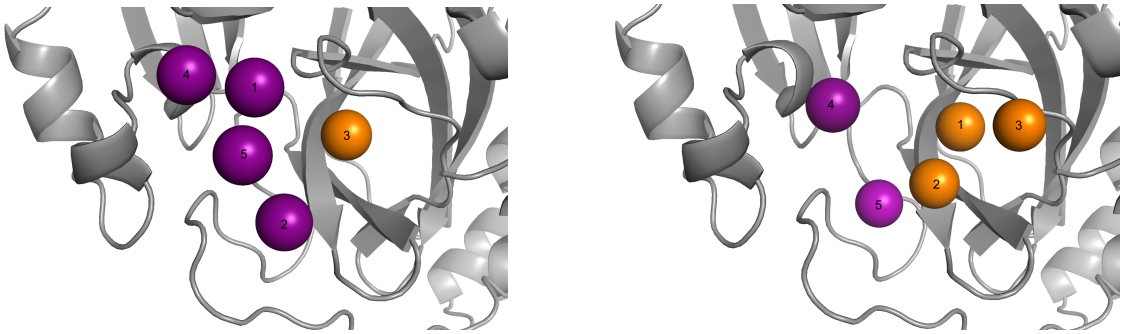
	PointVS		Gnina	
	PCC	ρ	PCC	ρ
Top 5	-0.710	0.640	-0.233	0.234
Top 10	-0.853	0.788	0.094	0.093

Table 4.4: Mean PCCs calculated between the scores of the top five and top ten ranked protein atoms by PointVS, and the distance between them and the nearest ligand atom. Also shown is the mean rank correlation calculated between these values, ρ. The means were calculated over a subset of 28 randomly selected bound structures from the PDBBind Core set (see Appendix B for PDB IDs).

rankings by PLIP and PointVS. When using Gnina, by contrast, we see a much weaker correlation for both of these tests.

These results verify that PointVS is capable of identifying intermolecular interactions in agreement with PLIP, providing further evidence that the edge attribution scores from PointVS are identifying important binding interactions.

4.3.5 Hotspot Identification using PointVS (carried out by Lucy Vost and Jack Scantlebury)



a b

Figure 4.5: Donor and acceptor hotspot maps in the binding pocket of Mpro, coloured purple and orange respectively, numbered according to their rank. The hotspots on the left were obtained with the Hotspots API, whilst the ones on the right were obtained with PointVS by performing attribution on the 142 structures of bound fragments available on Fragalysis.

Having shown that PointVS is potentially capable of identifying important binding interactions, we assess whether it can be used as a method for extracting structural information from a fragment screen, and hence used to guide molecule design. To do this, we extract hotspot maps from PointVS by performing edge attention attribution

on fragment screen data, and compare the resulting hotspots to those found using a data-driven approach, the Hotspots API [108].

Our first test took the fragment screen data available for the SARS-CoV-2 main protease (Mpro). The data was extracted from the Fragalysis platform [165]. A full list detailing the structures used is given in Appendix A. A comparison between the hotspots identified by PointVS attribution and the Hotspots API is shown in Fig 4.5. The hotspots of the two methods will not fall on exactly the same points, as the PointVS hotspots correspond to protein atoms, and the API hotspots to points in the binding pocket. Nevertheless, the two methods clearly highlight very separate regions of the binding pocket as important.

4.3.5.1 Dependency on Fragment Screen Size (carried out by Lucy Vost)

Given that PointVS identifies sites that are different from a traditional hotspot method, we next tested whether the high-scoring sites identified by PointVS are dependent on the input fragment set.

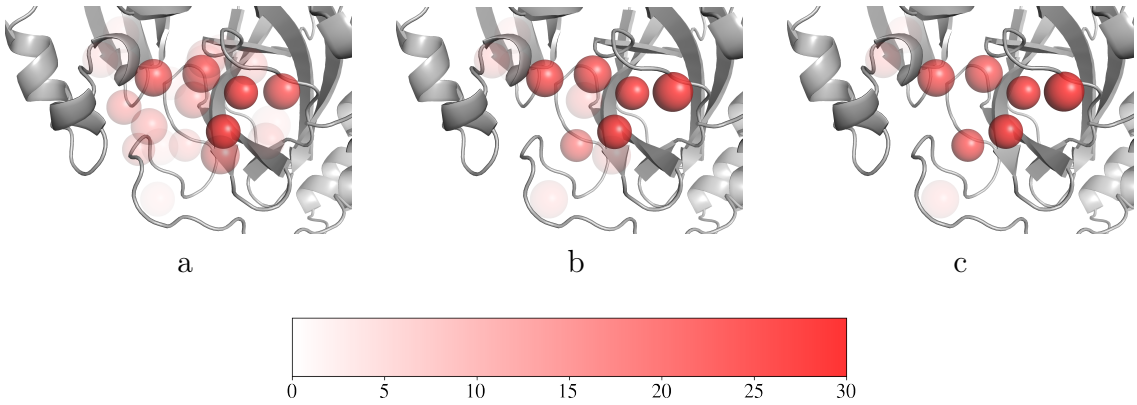


Figure 4.6: Top five scoring hotspot maps in the binding pocket of Mpro found by performing attribution on 40 sets of randomly sampled sets of 10 (a), 30 (b) and 80 (c) bound structures. Both donor and acceptor hotspots are shown in red. The spheres representing the hotspots are transparent and overlaid, so the more opaque a hotspot appears, the more times it was ranked in the top five.

To do this, we perform the same process as before, but instead of using all the fragment screen data (Appendix A) at once, we randomly sample smaller sets from it. We vary the size of the set generated between 10 and 80. We then take the top five highest scoring hotspots generated using PointVS with each randomly generated set of fragment hits. The results of this are shown in Fig 4.6, where we took samples of size 10, 30 and 80 from the available bound structures. This random sampling process was

carried out 40 times. We see that when the hotspot maps are extracted from a smaller number of bound structures, a wider range of protein atoms are identified as being important; PointVS is less able to consistently pick out the same five protein atoms every time (although we note that even when using the smallest sets of 10 bound structures, the five hotspots that are identified when using the full fragment screen are the most commonly included, hence their darker colour in Fig 4.6). Increasing the number of bound structures given to PointVS results in more consistent hotspot maps being generated.

4.3.6 Fragment Elaboration (carried out by Lucy Vost)

Having demonstrated that the PointVS hotspots are consistent amongst themselves, and that they highlight different areas in the Mpro binding site to the API hotspots, we assess whether they are highlighting important areas by performing fragment elaboration. If PointVS is successfully identifying important interactions, molecules generated to satisfy the pharmacophoric constraints imposed by our new hotspots should be as good, if not better, than the molecules generated with the API hotspots.

We performed attribution on the full set of Mpro crystal structures to obtain PointVS hotspot maps, and used the Hotspots API to obtain a second set of hotspot maps. We use each of the top ten ranked hotspots from each set of maps to elaborate on 109 fragments. The number of fragments that were successfully elaborated on using each hotspot is shown in Table 4.6, alongside the mean number of elaborations made per fragment.

To assess the binding affinity of the generated elaborations, we dock them using GOLD, as well as the fragment they were grown from, and use both docking scores to calculate a standardised ligand efficiency score, ΔSLE (see Section 4.2.4). We then rank the elaborations by these scores, and take the average of the top 20 scores to obtain ΔSLE_{20}. To see the performance of a hotspot over all fragments successfully elaborated with it, we take the mean of the ΔSLE_{20}s obtained.

Method	PointVS	API
Hotspots 1-5: ΔSLE_{20}	**0.304**	0.213
Hotspots 6-10: ΔSLE_{20}	0.146	**0.269**

Table 4.5: The change in the mean standardised ligand efficiency of the top 20 highest-scoring molecules (ΔSLE_{20}) generated using hotspots ranked 1-5 and 6-10 by PointVS and the Hotspots API for Mpro. The highest ΔSLE_{20} for both ranges of hotspots is shown in bold.

	PointVS		Hotspots API	
Rank	Fragments Elaborated	Elaborations per Fragment	Fragments Elaborated	Elaborations per Fragment
1	57	215	26	189
2	82	182	57	183
3	51	202	59	207
4	14	199	9	158
5	21	223	28	179
6	60	183	23	162
7	46	228	3	220
8	78	216	54	197
9	24	215	0	0
10	47	218	28	179

Table 4.6: The number of fragments that were successfully elaborated on using PointVS and API hotspots for Mpro, and the mean number of elaborations that were generated for each fragment. STRIFE was provided with 109 fragments to elaborate on for each hotspot, and asked to generate 250 elaborations for each one. Ranking for PointVS refers to the rank of the mean rank for that atom's edge attention score across structures.

As shown in Table 4.5, the highest scoring elaborations by our ΔSLE_{20} metric were generated by STRIFE using the top five PointVS hotspots as input. We find that the elaborations made using the hotspots ranked 6-10 by PointVS have far worse ΔSLE_{20}s. By contrast, the Hotspots API is successful in identifying ten high-scoring hotspots (indeed, on average, these top ten hotspots produce elaborations with higher changes in standardised ligand efficiencies than the PointVS hotspots), but is unable to successfully rank them amongst themselves.

It is generally important for hotspots to be accurately ranked, as in real-world fragment-to-lead campaigns, only a small number of them can easily be targeted.

To test whether PointVS hotspots are also reliable for other proteins, we repeat the above process for a number of other targets: SARS-CoV-2 non-structural protein 3 (mArh), SARS-CoV-2 non-structural protein 14 (NSP14), and macrophage-1 antigen (Mac1). These all have fragment screen datasets of various sizes available on Fragalysis [165] (see Appendix A). Each of the top ten hotspots produced by PointVS and the API were tested, the results of which are shown in Table 4.7. The targets mArh, NSP14 and Mac1 were tested on 35, 30 and 67, respectively, and the number of successfully elaborated fragments and mean number of elaborations generated per fragment are provided in Table 4.8

On average, the change in standardised ligand efficiency of molecules generated

	Mpro: 142		mArh: 61		NSP14: 19		Mac1: 58	
ΔSLE_{20}	PointVS	API	PointVS	API	PointVS	API	PointVS	API
Hotspots 1-5	**0.341**	0.213	**1.749**	1.581	**1.708**	1.240	0.707	**1.081**
Hotspots 6-10	0.189	**0.269**	1.727	**1.748**	**1.143**	1.193	0.740	**1.023**

Table 4.7: Mean standardised ligand efficiency of the top 20 elaborations (ΔSLE_{20}) made using PointVS hotspots and API hotspots on four different targets. The numbers after the target names refer to the number of bound fragment structures available for each target. The list of Fragalysis codes corresponding to the structures used is given in Appendix A. The highest ΔSLE_{20} for both ranges of hotspots is shown in bold.

with PointVS hotspot maps is greater than those generated with the API hotspots. We also see that the hotspots ranked 1-5 by PointVS usually outperform those ranked 6-10, suggesting that the attribution method is good not only at picking out important atoms, but also ranking them amongst themselves. This is not the case with the Hotspots API.

4.3.6.1 Case study (carried out by Lucy Vost)

To demonstrate the potential of the PointVS hotspot maps for real-world fragment-to-lead campaigns, we present a case study on a well-studied set of drug targets: the fibroglast growth factor receptors. Aberrant activation of these receptors is associated with a wide variety of cancers, making them a promising druggable target. In 2019, a collaboration between Astex and Newcastle University resulted in erdafitinib [168]: a drug that has since been FDA approved, making it the third fragment-derived drug to reach this stage.

In the process of designing erdafitinib, the collaborators heavily relied on a fragment screen performed on the target in 2008. A fragment found in this screen, shown in Fig 4.7, was found to have high potency, and was used as a starting point in the design process.

In Patani *et al.* [169], two hydrogen bonds are found to form between erdafitinib and the protein. We investigate whether we can identify the protein atoms forming these bonds as important using either PointVS or the API hotspots.

The API places one donor hotspot 1.5 Å away from the location of one of the erdaftnib atoms that forms a hydrogen bond with a protein atom. However, it does not recognise the point the second binding ligand atom is found at as important.

Mac1

	PointVS		Hotspots API	
Rank	Fragments successfully elaborated	Elaborations per fragment	Fragments successfully elaborated	Elaborations per fragment
1	21	217	15	189
2	12	187	10	183
3	13	223	11	207
4	11	192	7	158
5	5	207	12	179
6	9	176	15	162
7	6	239	21	220
8	2	222	11	197
9	0	0	0	0
10	1	218	8	210

NPS14

	PointVS		Hotspots API	
Rank	Fragments successfully elaborated	Elaborations per fragment	Fragments successfully elaborated	Elaborations per fragment
1	14	194	2	237
2	3	196	23	218
3	24	215	24	195
4	23	186	14	154
5	14	183	0	0
6	23	186	24	195
7	17	194	0	0
8	5	196	4	116
9	1	89	27	199
10	6	205	13	181

Table 4.8: (Table by Lucy Vost) The number of fragments that were successfully elaborated on using PointVS and API hotspots for Mac1 (top) and NSP14 (bottom), and the mean number of elaborations generated per fragment. Ranking for PointVS refers to the ranking of the mean edge attention for the atom associated with that hotspot across all input structures from the fragment screen. For Hotspots API, the ranking refers to the order of Hotspot API scores from the single input reference structure.

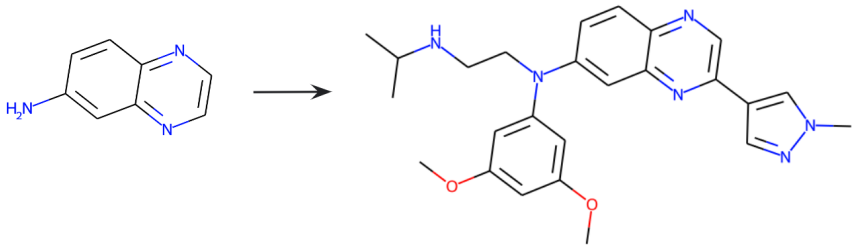


Figure 4.7: Left, the fragment used as a starting point in the development of erdafitinib, shown on the right.

To obtain the PointVS hotspots, we perform attribution on 16 of the bound structures of FHFR1 available in the PDB [68] (see Appendix C for PDB IDs), none of which have a Tanimoto similarity of greater than 0.7 to erdafitinib. We find that it correctly identifies both of the protein atoms that form hydrogen bonds with ligand atoms as important, ranking them as the third and sixth most important atoms for binding.

4.4 Conclusions (by Lucy Vost and Jack Scantlebury)

In this chapter we described the development of PointVS, a state-of-the-art EGNN based method for affinity prediction, and the first ML-based method for extracting important binding information from a target for molecule design.

During the development of PointVS we also identified that a commonly used benchmark for machine learning-based scoring functions, CASF-16, overestimates their accuracy when trained using the most commonly used training dataset. In light of this, PointVS was trained and tested on a rigorously filtered dataset, and hence encouraged to learn the rules governing intermolecular binding rather than memorise training data. We show that when trained on this filtered dataset, PointVS achieves comparable results to other leading scoring functions, providing some evidence that it is successfully identifying important binding interactions. We provide further proof of this by performing attribution, and show that PointVS is able to identify intermolecular interactions in line with those found by PLIP, a distance-based tool for profiling protein-ligand interactions.

Finally, we investigated how knowledge of important binding regions could be leveraged in other stages of the drug development pipeline; namely fragment elab-

oration. We show that when provided with a fragment screen of around 30 bound structures, performing attribution on such structures can yield hotspot maps describing important protein atoms. Using these in a fragment elaboration tool results in improved docking scores for the elaborated molecules compared to when hotspots are obtained from a data-based method. This provides further evidence that PointVS is not learning to memorise ligand information but to recognise important interactions, and additionally constitutes the first ML-based method of extracting structural information from a protein target in a way that is useful for fragment elaboration. More broadly, our work demonstrates that attribution techniques, when applied to debiased models, can be useful for extracting structural information in a way that can be useful in molecule generation.

4.5 Remarks on Chapter 4

The standard benchmark train/test split, into the PDBBind general set and the CASF-16 subset, is clearly not fit for purpose. Although this has been pointed out before [111, 147], frustratingly few models are trained and tested using the simple information leakage protection methods we employed. We suspect that this is because results are difficult to publish if they do not advance the 'headline figure' of Pearson's Correlation Coefficient or root mean-squared error on affinity values. As described in this chapter, we may have hit a ceiling in the information that machine learning models can encode about protein-ligand complex space from the very limited number of adequately labelled, accurate crystal structures.

4.5.1 Water and Dynamics

There are other limitations to this approach. Protein-ligand complexes are dynamic objects, often continuously moving between various local minima, each associated with a different ligand pose and protein conformation. The kinetics of an interaction are ignored entirely in training and testing, and entropic contributions are only accounted for implicitly, if at all. Water-ligand and water-protein interactions, as well as protein-ligand interactions mediated by water molecules, have been shown to be of vital importance in protein-ligand binding [170, 171, 172]. An obstacle to using water molecules in these machine learning-based scoring functions is that waters are constantly in flux. This is most true of surface waters, which exchange with bulk waters, but is also true of waters held more rigidly in place in polar cavities [173]. Any static, water-free image of a protein-ligand complex cannot capture the true energy landscape of the interaction.

4.5.2 Attribution Datasets

Input attribution can work in tandem with good benchmarking protocols to ensure that the physics of binding interactions is learned by ML-based SBVS algorithms, rather than classifying new protein-ligand complexes based on their similarity to ligands and proteins in the training set. In this chapter, we showed that the edge attention scores from PointVS can point to important intermolecular interactions. The evidence for this is still largely circumstantial. The fact that STRIFE gives a larger change in standardised ligand efficiency using hotspots generated using GNN

attention scores is further evidence that the ranking given by PointVS is correlated in some way with interaction importance; case studies on specific protein-ligand complexes with interactions drawn from careful comparative studies hold no statistical power.

What is required is a large-scale test dataset for different methods of attribution to be tested against. This could be a scaled-up version of the so-called 'magic methyl' experiments, where the addition of just a single methyl group [174, 175] causes a large change in binding affinity, perhaps due to the eviction of a water molecule from a hydrophobic pocket. Literature on these, and other activity cliffs, often lacks structural data for at least one of the two relevant binding events. Producing the large number of these structural pairs required for statistically sound attribution validation sets would be difficult.

Another solution could be all-atom quantum mechanical calculations of protein-ligand binding sites, or even whole proteins. Perturbations, such as removing certain atoms or functional groups from either molecule followed by some method of computational relaxation, could be used to approximate the contribution of those entities to overall affinity. This is computationally expensive; every functional group of interest in the ligand and protein would require its own calculation and accurate full atom quantum mechanical treatment of a complex scales with the cube of the number of atoms. However, density-functional theorem (DFT) studies of proteins are becoming more accessible on inexpensive hardware; there is precedent for using DFT to study protein-ligand complexes [176, 177, 178], and some faster methods of semi-empirical quantum mechanical methods are fast enough for this kind of large-scale study [179]. The previously mentioned problem of dynamics and water molecules would persist.

Chapter 5

Conclusion and Future Work

The main topic of this thesis has been the scoring of a static image of a ligand bound to a protein. The 'score' here references either the probability that the pose is close to what would be observed by crystallography, or the affinity with which a ligand in a specific pose would bind to a protein. A very early example of scoring protein-ligand complexes with neural networks came in 2010, when Durant *et al.* scored docked poses using hand-crafted features as input to a fully connected neural network [180]. An early use of a GNN for predicting whether or not ligands would bind to proteins uses the molecular graph of the ligand alongside the sequence of the protein, with no reference to Cartesian coordinates of atoms in either molecule [21]. 'Full-picture' SBVS, in which atoms and residues are explicitly used as input, began with CNNs such as Gnina [24, 111] and Pafnucy [64], and has progressed onto GNNs and especially $SE(3)$-invariant GNNs [63].

The plethora of ML methods for both pose ranking and affinity prediction have given modest gains in accuracy, as defined by measures such as correlation coefficients and ROC-AUC on standard validation sets such as the PDBBind core set. As mentioned in all of the previous chapters, care must be taken when constructing test and training sets to ensure that performance on test sets is a good indicator of performance on the space of all protein-ligand complexes. Unfortunately, it is difficult to compare performance of algorithms such as PointVS to others, as most publications do not take the important step of removing similar proteins and ligands from different splits. A major example of this is the numbers reported in the CASF benchmarking paper [48], against which many subsequent works are compared. The 'headline figure' of Pearson's Correlation Coefficient between measured and predicted affinities has crept up, but findings in Chapter 4 show that this is not a good measure of the

true accuracy of a scoring function.

Multiple publications, including my own, have pointed out these problems with common training and testing protocols [63, 74, 75, 106, 121, 147]. There are two factors which influence the generalisability of an ML-based scoring function: the model architecture, and the training data. I believe that the similar performance of multiple, distinct neural network architectures when trained and tested on the same data indicates that architecture is not the limiting factor. Higher accuracy than non-ML scoring functions such as AutoDock Vina will come from training on data that is more representative of protein-ligand space.

Tsubaki *et al.* claim that for chemical and biological sciences, 'interpolation' accuracy (accuracy on, for example, a held-out split of the training set) is not a good measure for generalisability of an ML model [181]. Instead, they posit that the measure of accuracy should be 'extrapolation' accuracy, or accuracy on externally constructed validation sets. These sets will have different distributions of properties. For example, an ML-based scoring function trained on large molecules should be able to predict the affinity or pose of fragments. This is supported by the findings in Chapters 2 and 4, where clustered cross-validation on DUD-E gave a more impressive picture of performance than on the ChEMBL validation set. The same argument can be made about the relationship between the PDBBind General and Core sets.

In lieu of sufficiently large, representative, and unbiased training datasets, we have shown in Chapter 2 that careful dataset augmentation can force deep neural networks to use information from all parts of the input. The models trained in this way also generalise better to externally constructed test sets, although performance still lags behind that seen on clustered cross-validation splits of the training set. A fair criticism of that work (and of the DUD-E dataset) is that it conflates two problems: the question of pose, and the question of binders/non-binders. The way the ChEMBL validation set was constructed and assessed, following Imrie *et al.* [23], further muddied the water, as an aggregate of several different poses was taken for the final score, which is not how the network was trained. These issues were subsequently addressed in Chapter 4, where pose classification and affinity prediction are treated as separate but related problems.

One way to verify that a deep neural network is learning the rules of binding as observed in nature is by using input attribution. As discussed in Chapter 4, what should be used as the ground truth against which to compare attribution scores is an open question. It is something which if solved would be a valuable resource in developing reliable, generalisable machine learning-based scoring functions. I believe the method I described at the end of Chapter 4 would be a viable, albeit expensive, way to generate such a dataset, which should be used to benchmark any new ML-based scoring functions.

The architecture of PointVS (Chapter 4) is a marked improvement on TransFS (Chapter 2). This is not because of its performance, which is only slightly better, but because it is easier to attribute importance to parts of the input. Besides improvements in the availability of more high quality training data, there are two architectural changes which may give a performance boost.

The first, which has been implemented in the code base but for which the required data has not been generated, is the inclusion of strain energy in training and inference. In its current implementation, the final pooled graph embedding is concatenated with the difference between the gas phase pose energy and the calculated energy when bound to the protein. This represents the steric clashes and torsional strain that are imposed on the ligand when binding occurs, and can be estimated by computationally relaxing the bound pose without the protein. It is well known that ligand strain is an important feature in binding. One downside of including this information is that it could impair the rest of the network from learning the physics of binding, which should include ligand strain implicitly.

The second is to expand on the idea of edge attention to using full self-attention (as described in [157]) on node features at the end of the network. Currently, attention is only applied to edges, and is a shallow multi-layer perceptron. Full self-attention means that the atom embeddings at the end would be linear combinations of all other atom embeddings, and the atom-atom interaction importance could be extracted from the attention matrix. As long as the usual addition of positional embeddings is avoided, the operation remains $SE(3)$-invariant.

I believe that ML-based scoring functions have advanced as far they can using publicly available datasets. The future lies with large-scale, high quality, publicly avail-

able ensembles of structures for a diverse set of ligands and proteins, complete with water molecules. Validation should be carried out carefully, and attribution should be used to prove that ligand-protein interactions are learned rather than dataset biases. Attribution datasets to this end should be produced, probably computationally, to aid future development.

References

[1] Christopher A. Lipinski et al. "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings". *Advanced Drug Delivery Reviews* 23.1 (1997). In Vitro Models for Selection of Development Candidates, pp. 3–25.

[2] Olivier J. Wouters, Martin McKee, and Jeroen Luyten. "Estimated Research and Development Investment Needed to Bring a New Medicine to Market, 2009-2018". *JAMA* 323.9 (Mar. 2020), pp. 844–853.

[3] Suzanne B. Shuker et al. "Discovering High-Affinity Ligands for Proteins: SAR by NMR". *Science* 274.5292 (1996), pp. 1531–1534.

[4] Qingxin Li. "Application of fragment-based drug discovery to versatile targets". *Frontiers in Molecular Biosciences* 7 (2020).

[5] John E. Ladbury, Gerhard Klebe, and Ernesto Freire. "Adding calorimetric data to decision making in lead discovery: a hot tip". *Nature Reviews Drug Discovery* 9.1 (2010), pp. 23–27.

[6] Andrew D. Scott et al. "Thermodynamic Optimisation in Drug Discovery: A Case Study using Carbonic Anhydrase Inhibitors". *ChemMedChem* 4.12 (2009), pp. 1985–1989.

[7] Caroline R. Buchholz and William C. K. Pomerantz. "19F NMR viewed through two different lenses: ligand-observed and protein-observed 19F NMR applications for fragment-based drug discovery". *RSC Chem. Biol.* 2 (5 2021), pp. 1312–1330.

[8] Stephan Niebling et al. "FoldAffinity: binding affinities from nDSF experiments". *Scientific Reports* 11.1 (2021), p. 9572.

[9] Andreas Lingel et al. "Comprehensive and High-Throughput Exploration of Chemical Space Using Broadband 19F NMR-Based Screening". *Angewandte Chemie International Edition* 59.35 (2020), pp. 14809–14817.

[10] Anandalakshmi Venkatraman et al. "Pharmaceutical modulation of the proteolytic profile of Transforming Growth Factor Beta induced protein (TGFBIp) offers a new avenue for treatment of TGFBI-corneal dystrophy". *Journal of Advanced Research* 24 (2020), pp. 529–543.

[11] Michael Saur et al. "Fragment-based drug discovery using cryo-EM". *Drug Discovery Today* 25.3 (2020), pp. 485–490.

[12] S Ekins, J Mestres, and B Testa. "In silico pharmacology for drug discovery: methods for virtual ligand screening and profiling". *British Journal of Pharmacology* 152.1 (2007), pp. 9–20.

[13] Jiankun Lyu et al. "Ultra-large library docking for discovering new chemotypes". *Nature* 566.7743 (2019), pp. 224–229.

[14] Joseph Rebehmed et al. "Editorial: Advances in Molecular Docking and Structure-Based Modelling". *Frontiers in Molecular Biosciences* 9 (2022).

[15] W. Patrick Walters and Renxiao Wang. "New Trends in Virtual Screening". *Journal of Chemical Information and Modeling* 59.9 (2019), pp. 3603–3604.

[16] Si-sheng Ou-Yang et al. "Computational drug discovery". *Acta Pharmacologica Sinica* 33.9 (2012), pp. 1131–1140.

[17] Markus Boehm et al. "Similarity Searching and Scaffold Hopping in Synthetically Accessible Combinatorial Chemistry Spaces". *Journal of Medicinal Chemistry* 51.8 (2008), pp. 2468–2480.

[18] Dawid Lichosyt et al. "The Influence of Binding Site Geometry on Anion-Binding Selectivity: A Case Study of Macrocyclic Receptors Built on the Azulene Skeleton". *Chemistry – A European Journal* 24.45 (2018), pp. 11683–11692.

[19] Malgorzata N. Drwal and Renate Griffith. "Combination of ligand- and structure-based methods in virtual screening". *Drug Discovery Today: Technologies* 10.3 (2013), e395–e401.

[20] Simone Sciabola et al. "Critical Assessment of State-of-the-Art Ligand-Based Virtual Screening Methods". *Molecular Informatics* 41.11 (2022). paper no. 2200103, p. 1.

[21] Masashi Tsubaki, Kentaro Tomii, and Jun Sese. "Compound–protein interaction prediction with end-to-end learning of neural networks for graphs and sequences". *Bioinformatics* 35.2 (2019), pp. 309–318.

[22] Sheila C. Araujo et al. "Structure-Based Virtual Screening, Molecular Dynamics and Binding Free Energy Calculations of Hit Candidates as ALK-5 Inhibitors". *Molecules* 25.2 (2020), p. 264.

[23] Fergus Imrie et al. "Protein Family-Specific Models Using Deep Neural Networks and Transfer Learning Improve Virtual Screening and Highlight the Need for More Data". *Journal of Chemical Information and Modeling* 58.11 (2018), pp. 2319–2330.

[24] Matthew Ragoza et al. "Protein-Ligand Scoring with Convolutional Neural Networks". *Journal of Chemical Information and Modeling* 57.4 (2017), pp. 942–957.

[25] Stephan Raub et al. "AIScore - Chemically Diverse Empirical Scoring Function Employing Quantum Chemical Binding Energies of Hydrogen-Bonded Complexes". *Journal of Chemical Information and Modeling* 48.7 (2008), pp. 1492–1510.

[26] Alejandro Varela-Rial, Maciej Majewski, and Gianni De Fabritiis. "Structure based virtual screening: Fast and slow". *WIREs Computational Molecular Science* 12.2 (2022), e1544.

[27] Oleg Trott and Arthur J. Olson. "AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading". *Journal of Computational Chemistry* 31.2 (2010), pp. 455–461.

[28] Claudio N. Cavasotto and M. Gabriela Aucar. "High-Throughput Docking Using Quantum Mechanical Scoring". *Frontiers in Chemistry* 8 (2020).

[29] J. Andrew McCammon, Bruce R. Gelin, and Martin Karplus. "Dynamics of folded proteins". *Nature* 267.5612 (1977), pp. 585–590.

[30] Bernard R. Brooks et al. "CHARMM: A program for macromolecular energy, minimization, and dynamics calculations". *Journal of Computational Chemistry* 4.2 (1983), pp. 187–217.

[31] Garrett M. Morris et al. "AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility". *Journal of Computational Chemistry* 30.16 (2009), pp. 2785–2791.

[32] Garrett M. Morris et al. "Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function". *Journal of Computational Chemistry* 19.14 (1998), pp. 1639–1662.

[33] David A. Case et al. "The Amber biomolecular simulation programs". *Journal of Computational Chemistry* 26.16 (2005), pp. 1668–1688.

[34] Gareth Jones et al. "Development and validation of a genetic algorithm for flexible docking". *Journal of Molecular Biology* 267.3 (1997), pp. 727–748.

[35] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". *Physical Review* 140.4A (1965), A1133–A1138.

[36] Michael M. Mysinger et al. "Directory of Useful Decoys, Enhanced (DUD-E): Better Ligands and Decoys for Better Benchmarking". *Journal of Medicinal Chemistry* 55.14 (2012), pp. 6582–6594.

[37] A. Ben-Naim. "Statistical potentials extracted from protein structures: Are these meaningful potentials?" *The Journal of Chemical Physics* 107.9 (1997), pp. 3698–3706.

[38] Hans F. G. Velec, Holger Gohlke, and Gerhard Klebe. "DrugScoreCSDKnowledge-Based Scoring Function Derived from Small Molecule Crystal Data with Superior Recognition Rate of Near-Native Ligand Poses and Better Affinity Prediction". *Journal of Medicinal Chemistry* 48.20 (2005), pp. 6296–6303.

[39] Ingo Muegge and Yvonne C. Martin. "A General and Fast Scoring Function for Protein-Ligand Interactions: A Simplified Potential Approach". *Journal of Medicinal Chemistry* 42.5 (1999), pp. 791–804.

[40] Qiancheng Shen et al. “Knowledge-Based Scoring Functions in Drug Design: 2. Can the Knowledge Base Be Enriched?” *Journal of Chemical Information and Modeling* 51.2 (2010), pp. 386–397.

[41] Hans-Joachim Böhm. “The development of a simple empirical scoring function to estimate the binding constant for a protein-ligand complex of known three-dimensional structure”. *Journal of Computer-Aided Molecular Design* 8.3 (1994), pp. 243–256.

[42] DASSAULT SYSTÈMES. *BIOVIA Discovery Studio*. San Diego: 2016.

[43] Renxiao Wang, Luhua Lai, and Shaomeng Wang. “Further development and validation of empirical scoring functions for structure-based binding affinity prediction”. *Journal of Computer-Aided Molecular Design* 16.1 (2002), pp. 11–26.

[44] Renxiao Wang et al. “The PDBbind Database: Collection of Binding Affinities for Protein-Ligand Complexes with Known Three-Dimensional Structures”. *Journal of Medicinal Chemistry* 47.12 (2004), pp. 2977–2980.

[45] John Baxter. “Local optima avoidance in depot location”. *Journal of The Operational Research Society* 32.9 (1981), pp. 815–819.

[46] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

[47] Nicholas Metropolis et al. “Equation of State Calculations by Fast Computing Machines”. *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

[48] Minyi Su et al. “Comparative Assessment of Scoring Functions: The CASF-2016 Update”. *Journal of Chemical Information and Modeling* 59.2 (2019), pp. 895–913.

[49] Cheng Wang and Yingkai Zhang. “Improving scoring-docking-screening powers of protein–ligand scoring functions using random forest”. *Journal of Computational Chemistry* 38.3 (2017), pp. 169–177.

[50] Richard A. Friesner et al. “Glide: A New Approach for Rapid, Accurate Docking and Scoring. 1. Method and Assessment of Docking Accuracy”. *Journal of Medicinal Chemistry* 47.7 (2004), pp. 1739–1749.

[51] Alejandro Castro-Alvarez, Anna Costa, and Jaume Vilarrasa. “The Performance of Several Docking Programs at Reproducing Protein-Macrolide-Like Crystal Structures”. *Molecules* 22.1 (2017), p. 136.

[52] “Web of Science Search”. *Web of Science, http://webofscience.com* (2022).

[53] Wendy L. Williams et al. “The Evolution of Data-Driven Modeling in Organic Chemistry”. *ACS Central Science* 7.10 (2021), pp. 1622–1637.

[54] James B. Jr. Dunbar et al. “CSAR Benchmark Exercise of 2010: Selection of the Protein–Ligand Complexes”. *Journal of Chemical Information and Modeling* 51.9 (2011), pp. 2036–2046.

[55] Liwei Li, Bo Wang, and Samy O. Meroueh. "Support Vector Regression Scoring of Receptor–Ligand Complexes for Rank-Ordering and Virtual Screening of Chemical Libraries". *Journal of Chemical Information and Modeling* 51.9 (2011), pp. 2132–2138.

[56] Saurabh Bundela, Anjana Sharma, and Prakash S Bisen. "Potential compounds for oral cancer treatment: resveratrol, nimbolide, lovastatin, bortezomib, vorinostat, berberine, pterostilbene, deguelin, andrographolide, and colchicine". *PLoS ONE* 10.11 (2015), e0141719.

[57] Shujun Huang et al. "Applications of Support Vector Machine (SVM) Learning in Cancer Genomics". *Cancer Genomics and Proteomics* 15.1 (2018).

[58] Leo Breiman. "Random forests". *Machine Learning* 45.1 (2001), pp. 5–32.

[59] Raziur Rahman et al. "Heterogeneity Aware Random Forest for Drug Sensitivity Prediction". *Scientific Reports* 7.1 (2017), p. 11347.

[60] Alex P. Lind and Peter C. Anderson. "Predicting drug activity against cancer cells by random forest models based on minimal genomic information and chemical properties". *PLoS ONE* 14.7 (July 2019), pp. 1–20.

[61] Kyoungyeul Lee, Minho Lee, and Dongsup Kim. "Utilizing random Forest QSAR models with optimized parameters for target identification and its application to target-fishing server". *BMC Bioinformatics* 18.16 (2017), p. 567.

[62] Yann Lecun and Yoshua Bengio. "Convolutional networks for images, speech, and time-series". English (US). *The handbook of brain theory and neural networks*. Ed. by M.A. Arbib. MIT Press, 1995.

[63] Jack Scantlebury et al. "Data Set Augmentation Allows Deep Learning-Based Virtual Screening to Better Generalize to Unseen Target Classes and Highlight Important Binding Interactions". *Journal of Chemical Information and Modeling* 60.8 (2020), pp. 3722–3730.

[64] Marta M Stepniewska-Dziubinska, Piotr Zielenkiewicz, and Pawel Siedlecki. "Development and evaluation of a deep learning model for protein-ligand binding affinity prediction". *Bioinformatics* 34.21 (2018), pp. 3666–3674.

[65] Zhihai Liu et al. "Forging the Basis for Developing Protein–Ligand Interaction Scoring Functions". *Accounts of Chemical Research* 50.2 (2017), pp. 302–309.

[66] Martin Simonovsky and Joshua Meyers. "DeeplyTough: Learning Structural Comparison of Protein Binding Sites". *Journal of Chemical Information and Modeling* 60.4 (2020), pp. 2356–2366.

[67] Rajiv Gandhi Govindaraj and Michal Brylinski. "Comparative assessment of strategies to identify similar ligand-binding pockets in proteins". *BMC Bioinformatics* 19.1 (2018), p. 91.

[68] Helen M. Berman et al. "The Protein Data Bank". *Nucleic Acids Research* 28.1 (Jan. 2000), pp. 235–242.

[69] Maurice Weiler et al. "3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data". *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018.

[70] Mikhail Volkov et al. "On the Frustration to Predict Binding Affinities from Protein–Ligand Structures with Deep Neural Networks". *Journal of Medicinal Chemistry* 65.11 (2022), pp. 7946–7958.

[71] Kazuyuki Hara, Daisuke Saitoh, and Hayaru Shouno. *Analysis of dropout learning regarded as ensemble learning*. 2017. arXiv preprint, arxiv:1706.06859.

[72] Alan H. Lipkus et al. "Structural Diversity of Organic Chemistry. A Scaffold Analysis of the CAS Registry". *The Journal of Organic Chemistry* 73.12 (2008), pp. 4443–4451.

[73] Jochen Sieg, Florian Flachsenberg, and Matthias Rarey. "In Need of Bias Control: Evaluating Chemical Data for Machine Learning in Structure-Based Virtual Screening". *Journal of Chemical Information and Modeling* 59.3 (2019), pp. 947–961.

[74] Izhar Wallach and Abraham Heifets. "Most Ligand-Based Classification Benchmarks Reward Memorization Rather than Generalization". *Journal of Chemical Information and Modeling* 58.5 (2018), pp. 916–932.

[75] Lieyang Chen et al. "Hidden bias in the DUD-E dataset leads to misleading performance of deep learning in structure-based virtual screening". *PLOS ONE* 14.8 (2019), pp. 1–22.

[76] Viet-Khoa Tran-Nguyen, Célien Jacquemard, and Didier Rognan. "LIT-PCBA: An Unbiased Data Set for Machine Learning and Virtual Screening". *Journal of Chemical Information and Modeling* 60.9 (2020), pp. 4263–4273.

[77] Franco Scarselli et al. "The Graph Neural Network Model". *IEEE Transactions on Neural Networks* 20.1 (2009), pp. 61–80.

[78] Thomas N. Kipf and Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. 2016. arXiv preprint, arxiv:1609.02907.

[79] Jie Zhou et al. *Graph Neural Networks: A Review of Methods and Applications*. 2018. arXiv preprint, arxiv:1812.08434.

[80] Floris Geerts, Filip Mazowiecki, and Guillermo A. Pérez. *Let's Agree to Degree: Comparing Graph Convolutional Networks in the Message-Passing Framework*. 2020. arXiv preprint, arxiv:2004.02593.

[81] David K Duvenaud et al. "Convolutional Networks on Graphs for Learning Molecular Fingerprints". *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015.

[82] Justin Gilmer et al. *Neural Message Passing for Quantum Chemistry*. 2017. arXiv preprint, arxiv:1704.01212.

[83] Taco S. Cohen and Max Welling. *Group Equivariant Convolutional Networks*. 2016. arXiv preprint, arxiv:1602.07576.

[84] Benjamin Bloem-Reddy and Yee Whye Teh. *Probabilistic symmetries and invariant neural networks*. 2019. arXiv preprint, arxiv:1901.06082.

[85] Marc Finzi et al. *Generalizing Convolutional Neural Networks for Equivariance to Lie Groups on Arbitrary Continuous Data*. 2020. arXiv preprint, arxiv: 2002.12880.

[86] Fabian B. Fuchs et al. *SE(3)-Transformers: 3D Roto-Translation Equivariant Attention Networks*. 2020. arXiv preprint, arxiv:2006.10503.

[87] Michael J Hutchinson et al. "Lietransformer: equivariant self-attention for lie groups". *International Conference on Machine Learning*. PMLR. 2021, pp. 4533–4543.

[88] Raghunathan Ramakrishnan et al. "Quantum chemistry structures and properties of 134 kilo molecules". *Scientific Data* 1.1 (2014). paper no. 140022, p. 1.

[89] Lars Ruddigkeit et al. "Enumeration of 166 Billion Organic Small Molecules in the Chemical Universe Database GDB-17". *Journal of Chemical Information and Modeling* 52.11 (2012), pp. 2864–2875.

[90] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. *E(n) Equivariant Graph Neural Networks*. 2021. arXiv preprint, arxiv:2102.09844.

[91] Andrea Galassi, Marco Lippi, and Paolo Torroni. *Attention, please! A Critical Review of Neural Attention Models in Natural Language Processing*. 2019. arXiv preprint, arxiv:1902.02181.

[92] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv preprint, arxiv: 1706.03762.

[93] Steven Walczak. "Predicting crime and other uses of neural networks in police decision making". *Frontiers in Psychology* 12 (2021), p. 1.

[94] Annette Vestby and Jonas Vestby. "Machine Learning and the Police: Asking the Right Questions". *Policing: A Journal of Policy and Practice* 15.1 (June 2019), pp. 44–58.

[95] Vendy Fialková et al. "LibINVENT: reaction-based generative scaffold decoration for in silico library design". *Journal of Chemical Information and Modeling* 62.9 (2021), pp. 2046–2063.

[96] Fergus Imrie et al. "Deep Generative Models for 3D Linker Design". *Journal of Chemical Information and Modeling* 60.4 (2020), pp. 1983–1995.

[97] Fergus Imrie et al. "Deep generative design with 3D pharmacophoric constraints". *Chemical Science* 12.43 (2021), pp. 14577–14589.

[98] Stanislaw Antol et al. *VQA: Visual Question Answering*. 2015. arXiv preprint, arxiv:1505.00468.

[99] Sebastian Bach et al. "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation". *PLoS ONE* 10.7 (July 2015), pp. 1–46.

[100] Saurabh Desai and Harish G. Ramaswamy. "Ablation-CAM: Visual Explanations for Deep Convolutional Network via Gradient-free Localization". *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2020, pp. 972–980.

[101] Aditya Chattopadhay et al. "Grad-CAM+: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks". *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018.

[102] Bolei Zhou et al. *Learning Deep Features for Discriminative Localization*. 2015. arXiv preprint, arxiv:1512.04150.

[103] Joshua Hochuli et al. "Visualizing convolutional neural network protein-ligand scoring". *Journal of Molecular Graphics and Modelling* 84 (2018), pp. 96–108.

[104] Benjamin Sanchez-Lengeling et al. "Evaluating Attribution for Graph Neural Networks". *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 5898–5910.

[105] Thomas E. Hadfield, Jack Scantlebury, and Charlotte M. Deane. *Exploring The Ability Of Machine Learning-Based Virtual Screening Models To Identify The Functional Groups Responsible For Binding*. 2023. bioRxiv preprint, bioRxiv: 2023.04.29.538820.

[106] Jack Scantlebury et al. "A Small Step Toward Generalizability: Training a Machine Learning Scoring Function for Structure-Based Virtual Screening". *Journal of Chemical Information and Modeling* 63.10 (2023), pp. 2960–2974.

[107] Thomas E. Hadfield et al. "Incorporating Target-Specific Pharmacophoric Information into Deep Generative Models for Fragment Elaboration". *Journal of Chemical Information and Modeling* 62.10 (2022), pp. 2280–2292.

[108] Peter R Curran et al. "Hotspots api: a python package for the detection of small molecule binding hotspots and application to structure-based drug design". *Journal of Chemical Information and Modeling* 60.4 (2020), pp. 1911–1916.

[109] Colin R Groom et al. "The Cambridge structural database". *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials* 72.2 (2016), pp. 171–179.

[110] Irina Kufareva, Andrey V. Ilatovskiy, and Ruben Abagyan. "Pocketome: An encyclopedia of small-molecule binding sites in 4D". *Nucleic Acids Research* 40.D1 (2011), pp. D535–D540.

[111] Andrew T. McNutt et al. "GNINA 1.0: molecular docking with deep learning". *Journal of Cheminformatics* 13.1 (2021).

[112] Aleix Gimeno et al. "The light and dark sides of virtual screening: what is there to know?" *International Journal of Molecular Sciences* 20.6 (2019), p. 1375.

[113] Antonio Lavecchia and Carmen Di Giovanni. "Virtual screening strategies in drug discovery: a critical review". *Current Medicinal Chemistry* 20.23 (2013), pp. 2839–2860.

[114] Zhe Wang et al. "Comprehensive evaluation of ten docking programs on a diverse set of protein–ligand complexes: the prediction accuracy of sampling power and scoring power". *Physical Chemistry Chemical Physics* 18.18 (2016), pp. 12964–12975.

[115] Kristy A Carpenter and Xudong Huang. "Machine Learning-based virtual screening and its applications to Alzheimer's drug discovery: a review". *Current Pharmaceutical Design* 24.28 (2018), pp. 3347–3358.

[116] Adam Gonczarek et al. "Interaction prediction in structure-based virtual screening using deep learning". *Computers in Biology and Medicine* 100 (2018), pp. 253–258.

[117] Janaina Cruz Pereira, Ernesto Raul Caffarena, and Cicero Nogueira Dos Santos. "Boosting docking-based virtual screening with deep learning". *Journal of Chemical Information and Modeling* 56.12 (2016), pp. 2495–2506.

[118] Jessica Vamathevan et al. "Applications of machine learning in drug discovery and development". *Nature Reviews Drug Discovery* 18.6 (2019), pp. 463–477.

[119] Ingoo Lee, Jongsoo Keum, and Hojung Nam. "DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences". *PLoS Computational Biology* 15.6 (2019), e1007129.

[120] Han Altae-Tran et al. "Low data drug discovery with one-shot learning". *ACS Central Science* 3.4 (2017), pp. 283–293.

[121] Fergus Boyles, Charlotte M Deane, and Garrett M Morris. "Learning from the ligand: using ligand-based features to improve binding affinity prediction". *Bioinformatics* 36.3 (2020), pp. 758–764.

[122] Yann LeCun et al. "Gradient-based learning applied to document recognition". *Proceedings of The IEEE* 86.11 (1998), pp. 2278–2324.

[123] Amr H Mahmoud et al. "Elucidating the multiple roles of hydration for accurate protein-ligand binding prediction via deep learning". *Communications Chemistry* 3.1 (2020), pp. 1–13.

[124] AR Leach. "The ChEMBL database in 2017". *Nucleic Acids Research* 45 (2017), pp. D945–D954.

[125] Sabina Smusz, Rafał Kurczab, and Andrzej J Bojarski. "The influence of the inactives subset generation on the performance of machine learning methods". *Journal of Cheminformatics* 5.1 (2013), pp. 1–8.

[126] Vikram Sundar and Lucy Colwell. "Debiasing Algorithms for Protein Ligand Binding Data do not Improve Generalisation" (2019).

[127] Fergus Imrie, Anthony R Bradley, and Charlotte M Deane. "Generating property-matched decoy molecules using deep learning". *Bioinformatics* 37.15 (Feb. 2021), pp. 2134–2141.

[128] Kathrin Heikamp and Jurgen Bajorath. "Large-scale similarity search profiling of ChEMBL compound data sets". *Journal of Chemical Information and Modeling* 51.8 (2011), pp. 1831–1839.

[129] Jocelyn Sunseri and David R Koes. "Libmolgrid: graphics processing unit accelerated molecular gridding for deep learning applications". *Journal of Chemical Information and Modeling* 60.3 (2020), pp. 1079–1084.

[130] *OpenEye Toolkits*.

[131] James JP Stewart. "A method for predicting individual residue contributions to enzyme specificity and binding-site energies, and its application to MTH1". *Journal of Molecular Modeling* 22.11 (2016), pp. 1–19.

[132] Demetres D. Leonidas et al. "High-resolution crystal structures of ribonuclease A complexed with adenylic and uridylic nucleotide inhibitors. Implications for structure-based design of ribonucleolytic inhibitors". *Protein Science* 12.11 (2003), pp. 2559–2574.

[133] Cara L. Jenkins et al. "Binding of non-natural 3'-nucleotides to ribonuclease A". *The FEBS Journal* 272.3 (2005), pp. 744–755.

[134] Takaya Saito and Marc Rehmsmeier. "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets". *PLoS ONE* 10.3 (2015), e0118432.

[135] Angela Lopez-del Rio et al. "Evaluation of Cross-Validation Strategies in Sequence-Based Binding Prediction Using Deep Learning". *Journal of Chemical Information and Modeling* 59.4 (2019). PMID: 30730731, pp. 1645–1657.

[136] Kathrin Heikamp and Jürgen Bajorath. "Large-Scale Similarity Search Profiling of ChEMBL Compound Data Sets". *Journal of Chemical Information and Modeling* 51.8 (2011). PMID: 21728295, pp. 1831–1839.

[137] Michael M. Bronstein et al. *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. 2021. arXiv preprint, arxiv:2104.13478.

[138] Schrödinger, LLC. "The PyMOL Molecular Graphics System, Version 1.8". 2015.

[139] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020.

[140] *Anaconda Software Distribution*. Version Vers. 2-2.4.0. 2020.

[141] Sebastian Salentin et al. "PLIP: fully automated protein–ligand interaction profiler". *Nucleic Acids Research* 43.W1 (Apr. 2015), W443–W447.

[142] Deepak Vohra. "Apache Parquet". *Practical Hadoop Ecosystem: A Definitive Guide to Hadoop-Related Frameworks and Tools*. Berkeley, CA: Apress, 2016, pp. 325–335.

[143] Arthur Dalby et al. "Description of several chemical structure file formats used by computer programs developed at Molecular Design Limited". *Journal of Chemical Information and Computer Sciences* 32.3 (1992), pp. 244–255.

[144] wwPDB. *Atomic Coordinate Entry Format Description*. 2011.

[145] Geoff Hutchison et al. *openbabel/openbabel: Open Babel 3.1.0*. Version openbabel-3-1-0. Oct. 2016.

[146] Holger Krekel et al. *pytest*. 2004.

[147] Mikhail Volkov et al. "On the Frustration to Predict Binding Affinities from Protein–Ligand Structures with Deep Neural Networks". *Journal of Medicinal Chemistry* 65.11 (2022), pp. 7946–7958.

[148] Jiaqi Han et al. *Geometrically Equivariant Graph Neural Networks: A Survey*. 2022. arXiv preprint, arxiv:2202.07230.

[149] Chao Shen et al. "From machine learning to deep learning: Advances in scoring functions for protein–ligand docking". *WIREs Computational Molecular Science* 10.1 (2020), e1429.

[150] Derek Jones et al. "Improved Protein–Ligand Binding Affinity Prediction with Structure-Based Deep Fusion Inference". *Journal of Chemical Information and Modeling* 61.4 (2021), pp. 1583–1592.

[151] Jason Yosinski et al. "Understanding neural networks through deep visualization". *arXiv Preprint Arxiv:1506.06579* (2015).

[152] Wojciech Samek et al. "Evaluating the visualization of what a deep neural network has learned". *IEEE Transactions on Neural Networks and Learning Systems* 28.11 (2016), pp. 2660–2673.

[153] Liangzhen Zheng, Jingrong Fan, and Yuguang Mu. "OnionNet: a Multiple-Layer Intermolecular-Contact-Based Convolutional Neural Network for Protein–Ligand Binding Affinity Prediction". *ACS Omega* 4.14 (2019), pp. 15956–15965.

[154] Yanjun Li et al. "DeepAtom: A Framework for Protein-Ligand Binding Affinity Prediction". *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2019, pp. 303–310.

[155] Evan N Feinberg et al. "PotentialNet for molecular property prediction". *ACS Central Science* 4.11 (2018), pp. 1520–1530.

[156] Joseph A. Morrone et al. "Combining Docking Pose Rank and Structure with Deep Learning Improves Protein–Ligand Binding Mode Prediction over a Baseline Docking Approach". *Journal of Chemical Information and Modeling* 60.9 (2020), pp. 4170–4179.

[157] Ashish Vaswani et al. "Attention is All you Need". *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[158] Paul G. Francoeur et al. "Three-Dimensional Convolutional Neural Networks and a Cross-Docked Data Set for Structure-Based Drug Design". *Journal of Chemical Information and Modeling* 60.9 (2020), pp. 4200–4215.

[159] Irina Kufareva, Andrey V. Ilatovskiy, and Ruben Abagyan. "Pocketome: an encyclopedia of small-molecule binding sites in 4D". *Nucleic Acids Research* 40.D1 (2011), pp. D535–D540.

[160] Greg Landrum et al. *RdKit*. Version Release_2021_09_4. Jan. 2022.

[161] Jincai Yang, Cheng Shen, and Niu Huang. "Predicting or Pretending: Artificial Intelligence for Protein-Ligand Interactions Lack of Sufficiently Large and Unbiased Datasets". *Frontiers in Pharmacology* 11 (2020).

[162] Chris J Radoux et al. "Identifying interactions that determine fragment binding at protein hotspots". *Journal of Medicinal Chemistry* 59.9 (2016), pp. 4314–4325.

[163] Marcel L. Verdonk, Jason C. Cole, and Robin Taylor. "SuperStar: A Knowledge-based Approach for Identifying Interaction Sites in Proteins". *Journal of Molecular Biology* 289.4 (1999), pp. 1093–1108.

[164] Colin R. Groom et al. "The Cambridge Structural Database". *Acta Crystallographica Section B* 72.2 (2016), pp. 171–179.

[165] R. Skyner and F von Delft. *Xchem, Fragalysis*. https://fragalysis.diamond.ac.uk/. Accessed: 2022-07-30.

[166] Cheng Wang and Yingkai Zhang. "Improving scoring-docking-screening powers of protein–ligand scoring functions using random forest". *Journal of Computational Chemistry* 38.3 (2017), pp. 169–177.

[167] Jianing Lu et al. "Incorporating Explicit Water Molecules and Ligand Conformation Stability in Machine-Learning Scoring Functions". *Journal of Chemical Information and Modeling* 59.11 (2019), pp. 4540–4549.

[168] Christopher W. Murray, David R. Newell, and Patrick Angibaud. "A successful collaboration between academia, biotech and pharma led to discovery of erdafitinib, a selective FGFR inhibitor recently approved by the FDA". *Med. Chem. Commun.* 10 (2019), pp. 1509–1511.

[169] Harshnira Patani et al. "Landscape of activating cancer mutations in FGFR kinases and their differential responses to inhibitors in clinical use". *Oncotarget* 7.17 (2016), pp. 24252–24268.

[170] Balázs Zoltán Zsidó and Csaba Hetényi. "The role of water in ligand binding". *Current Opinion in Structural Biology* 67 (2021), 1—8.

[171] Johannes Schiebel et al. "Intriguing role of water in protein-ligand binding studied by neutron crystallography on trypsin complexes". *Nature Communications* 9.1 (2018), p. 3559.

[172] Di Cui et al. "The Role of Interfacial Water in Protein–Ligand Binding: Insights from the Indirect Solvent Mediated Potential of Mean Force". *Journal of Chemical Theory and Computation* 14.2 (2018), pp. 512–526.

[173] Oliviero Carugo. "Statistical survey of the buried waters in the Protein Data Bank". *Amino Acids* 48.1 (2015), 193–202.

[174] Heike Schönherr and Tim Cernak. "Profound Methyl Effects in Drug Discovery and a Call for New C-H Methylation Reactions". *Angewandte Chemie International Edition* 52.47 (2013), pp. 12256–12267.

[175] Cheryl S. Leung et al. “Methyl Effects on Protein-Ligand Binding”. *Journal of Medicinal Chemistry* 55.9 (2012), pp. 4489–4500.

[176] Carme Rovira. “Study of Ligand–Protein Interactions by Means of Density Functional Theory and First-Principles Molecular Dynamics”. *Methods in Molecular Biology (Clifton, N.J.)* 305 (Feb. 2005), pp. 517–54.

[177] Stephen J. Fox et al. “Density functional theory calculations on entire proteins for free energies of binding: Application to a model polar binding site”. *Proteins: Structure, Function, and Bioinformatics* 82.12 (2014), pp. 3335–3346.

[178] Lennart Gundelach et al. “Protein–ligand free energies of binding from full-protein DFT calculations: Convergence and choice of exchange–correlation functional”. *Physical Chemistry Chemical Physics* 23.15 (2021), 9381–9393.

[179] Nusret Yilmazer and Martin Korth. “Recent progress in treating protein–ligand interactions with quantum-mechanical methods”. *International Journal of Molecular Sciences* 17.5 (2016), p. 742.

[180] Jacob D. Durrant and J. Andrew McCammon. “NNScore: A Neural-Network-Based Scoring Function for the Characterization of Protein-Ligand Complexes”. *Journal of Chemical Information and Modeling* 50.10 (2010), pp. 1865–1871.

[181] Masashi Tsubaki and Teruyasu Mizoguchi. “On the equivalence of molecular graph convolution and molecular wave function with poor basis set”. *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1982–1993.

Appendix A

(Table by Lucy Vost) Fragalysis identification codes of the structures supplied to PointVS to obtain hotspots

Mpro			Mac1		NSP14
x10474_0A	x10484_0A	x10494_0A	DLS-EU0569_0A	DLS-X0626_0A	x1203_0D
x10898_0A	P0906_0B	x11543_0A	DLS-X0724_0A	DLS-X0689_0A	x1088_0D
x2964_0A	x11426_0A	x2646_0A	DLS-X0591_0A	DLS-X0711_0A	x1201_0D
x11372_0A	x10604_0A	x10559_0A	DLS-X0681_0A	DLS-EU0034_0A	x1316_0D
x11233_0A	x12682_0A	x11764_0A	DLS-X0895_1A	DLS-EU0087_0A	x1448_0D
x10473_0A	x10889_0A	x2971_0A	DLS-X0910_0A		x1412_0D
x10327_0A	P1470_0A	x11159_0A	DLS-X0587_0A		x1181_0D
x12740_0A	x10723_0A	x2908_0A	DLS-X0421_0A		x1199_0D
x10888_0A	x10248_0A	x11532_0A	DLS-EU0342_0A		x1197_0D
x10371_0A	P0045_0B	x0678_0A	DLS-X0844_0A		x1384_1D
x10606_0A	x10756_0A	x11164_0A	DLS-EU0815_0A		x1486_0D
x11743_0A	x10995_0A	P0045_0A	DLS-EU0291_0A		x1245_0D
x2912_0A	x10555_0A	x3108_0A	DLS-X0299_0A		x1541_0D
x10565_0A	x2572_0A	x11339_0A	DLS-X0471_0A		x1603_0D
x10422_0A	P0906_0A	x11564_0A	DLS-X0895_0A		x1236_2D
x10421_0A	P2007_0B	x10789_0A	DLS-EU0238_0A		x1282_0D
P1073_0A	x3359_0A	P2001_0B	DLS-X0766_0A		x1641_0D
x10488_0A	x10387_0A	x2608_0A	DLS-X0926_0A		x1078_0D
x11427_0A	x0107_0A	x10856_0A	DLS-EU0056_0A		x1212_1D
P0145_0B	x10201_0A	x10314_0A	DLS-X0592_0A		
x10942_0A	x11271_0A	x11513_0A	DLS-X0967_0A		
x10022_0A	x2581_0A	x11540_0A	DLS-EU0144_0A		
x11473_0A	x10377_0A	x2600_0A	DLS-X0722_0A		
x0434_0A	x10395_0A	x10598_0A	DLS-EU0704_0A		
P1470_0B	x10329_0A	x11041_0A	DLS-EU0844_0A		
x12719_0A	x10178_0A	x11417_0A	DLS-EU0481_0A		
x10396_0A	x12300_0A	x10638_0A	DLS-X0969_0A		
z7vh8_0A	x10324_0A	x10334_0A	DLS-EU0241_0A		
x11368_0A	P0884_0A	x10392_0A	DLS-X0962_0B		
P2007_0A	x11346_0A	x11708_0A	DLS-X0742_0A		
x11231_0A	x2649_0A	x11318_0A	DLS-EU0172_0A		
x10906_0A	x11641_0A	x10019_0A	DLS-X0918_0A		
x3080_0A	x10800_0A	x11428_0A	DLS-X1018_0A		
x11560_0A	x10996_0A	x10733_0A	DLS-X0962_0A		
x11013_0A	x10417_0A	x11475_0A	DLS-X0900_0A		
x10566_0A	x10247_0A	x10478_0A	DLS-X0685_0A		
P2001_0A	x3298_0A	x11485_0A	DLS-X0962_1A		
x11809_0A	x3366_0A	x10575_0A	DLS-X0805_0A		
x10679_0A	x2643_0A	P0145_0A	DLS-X0142_0A		
x10801_0A	x11488_0A	x10787_0A	DLS-X0104_0A		
P0925_0A	x11317_0A	x2569_0A	DLS-X0852_0B		
x11225_0A	x10889_1A	x11044_0A	DLS-X0918_1A		
x10338_0A	x10476_0A	x11723_0A	DLS-EU0716_0A		
x10359_0A	x11562_0A	x10322_0A	DLS-EU0123_0A		
x2193_0A	x10900_0A	x11354_0A	DLS-X0905_0A		
x10423_0A	x2562_0A	x12321_0A	DLS-X0910_1A		
x11557_0A	x11801_0A	x11025_0A	DLS-X0933_0A		
x10645_0A	x10237_0A	x10834_0A	DLS-X0837_0A		
x10626_0A	x11011_0A	x11507_0A	DLS-EU0293_0A		
x10236_0A	x10535_0A	x10976_0A	DLS-EU0424_0A		
x11186_0A	x10610_0A		DLS-EU0046_0A		

Appendix B

(By Lucy Vost) PDB IDs of randomly selected subset of 20 structures from the PDBBind Core set used in the calculation of the statistics in Table 4.4

1E66, 1H23, 1Y6R, 2BRB, 2FVD, 2QE4, 2VKM, 2XII, 2ZB1, 3AO4, 3GE7, 3OE5, 3RR4, 3UI7, 4DE1, 4DLI, 4JSZ, 4K18, 4M0Z, 4MGD

Appendix C

(By Lucy Vost) 16 of the bound structures of FHFR1 available in the PDB used for masking in Section 4.3.6.1

4NKS, 4NK9, 1AGW, 3C4F, 4NKA, 4V01, 4V05, 5B7V, 5A46, 4UWB 3RHX, 1FGI, 5A4C, 4UWC, 5VND, 6P69

Appendix D

Other Scripts

Several other scripts form part of the PointVS repository, but the work associated with them is not included in this thesis either because it was an avenue of inquiry which proved fruitless, or because the scripts were written for use by other researchers in projects that are not relevant to my work. These include:

- `for_steph.py` Script which takes text file containing locations of ligand and protein files, and outputs input attribution scores to atoms and bonds (for use by another student).
- `constrained_attribution.py` Script for comparing attribution scores across multiple bound ligands with common substructures. The pairwise changes in attribution scores for each ligand atom is compared with their relative displacement between bound structures, which could be obtained using constrained docking.
- `strain_energy.py` Uses RDKit [160] to estimate the internal strain energy of ligand poses, the idea being that the energy penalty associated with a given ligand pose should be a useful feature when predicting pose and affinity. The script works; the time to explore models incorporating this knowledge did not materialise.
- `gromacs.py` Takes molecular dynamics trajectories as pairwise protein-ligand atomic distances as a CSV, and plots variance and mean distance vs edge attribution scores of the crystal pose. The initial concept was to use the variance in atom-atom distance as a ground truth for the importance of noncovalent interactions, against which to compare GNN input attribution scores.