TU WIEN Informatics

Antibody-Antigen binding affinity prediction through the use of geometric deep learning

A framework for binding affinity prediction with Graph Neural Networks

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Fabian Traxler
Registration Number 01553958

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.-Prof. Dipl.-Ing.(BA) Dr.rer.nat. Thomas Gärtner, MSc
Assistance: Moritz Schäfer, Phd
Univ.-Ass. Dipl.-Ing. David Penz, B.A.
Univ.-Prof. Dr. Christoph Bock

Vienna, 10th January, 2023

Fabian Traxler | Thomas Gärtner

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Fabian Traxler

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 10. Jänner 2023

Fabian Traxler

Acknowledgements

First, I would like to thank Christoph Bock and Moritz Schäfer for providing me with the opportunity to be part of an exceptional team at the Bock Lab during my master's thesis. In particular, I am very grateful for all the support and discussion with Moritz Schäfer that not only guided me during my project but also provided valuable insights and countless life lessons. Furthermore, I would like to thank all people at the Bock Lab for taking me in so warm-heartedly and supporting me whenever necessary. My thanks also extend to the Medical University for providing me with the necessary computing resources to perform my experiments.

I am also grateful for my supervisors at TU Wien, Thomas Gärtner and David Penz, who were always open for discussions about the more technical aspects of the work and provided valuable feedback.

Finally, I want to express my thanks to my family and friends that not only enabled me to pursue my academic career but also sparked my initial interest in science.

Kurzfassung

Antikörper sind ein wesentlicher Bestandteil unseres Immunsystems, indem sie Immunreaktionen auslösen oder körperfremde Proteine unwirksam machen. Wissen über den Aufbau von Antikörper und deren konkreten Bindungsmechanismen erlaubt es, vielversprechende Antikörper für Therapien auszuwählen oder sogar neue Antikörper zu entwickeln. Obwohl einige regelbasierte Methoden (z.B. Kraftfelder) für diese Aufgabe adaptiert wurden, bleibt die Herausforderung, die Bindungsaffinität zwischen Antikörpern und Antigenen genau vorherzusagen, bestehen.

In dieser Arbeit schlagen wir einen rein datengetriebenen Ansatz zur Vorhersage von Antikörper-Antigen-Bindungsaffinität durch geometrische Deep-Learning-Methoden vor. Das Ziel unserer Arbeit ist es, die Leistung eines Graph-Neuronalen-Netzwerks (GNN) zu bewerten und es mit einem modernen Kraftfeld zu vergleichen. Dazu werden kristallisierte Antikörper-Antigen-Komplexe in Graphen Strukturen umgewandelt, um einen für maschinelles Lernen geeigneten Datensatz zu erstellen. Darüber hinaus werden ähnliche Daten (z.B. Protein-Protein-Komplexe) zusammengestellt, um die Auswirkungen des Transferlernens auf die Vorhersage der Antikörper-Antigen-Bindungsaffinität zu beurteilen.

Die Implementierung des von uns entwickelten GNN bietet ein PyTorch-Gerüst für die generische Vorhersage der Bindungsaffinität von graphähnlicher Strukturen. Das trainierte GNN zeigt vielversprechende Ergebnisse bei diversen Antikörper-Antigen-Daten und übertrifft das verglichene Kraftfeld. Die implementierten Transfer-Learning-Techniken führten nicht zu einer signifikanten Leistungsverbesserung, obwohl es noch zahlreiche solcher Techniken noch zu erforchen gibt. Die Effektivität von GNNs und ihre durchgängige Differenzierbarkeit unterstreichen ihr Potenzial für die Untersuchung von Antikörper-Bindung. Darüber hinaus ermöglichen diese Eigenschaften auch Anwendungen zur Verbesserung und dem Design von neuartigen experimentellen und therapeutischen Antikörpern.

Abstract

Antibodies are an integral part of our body's immune system due to their ability to trigger immune responses or render exogenous proteins ineffective. The mechanism of binding of antibodies has been studied thoroughly in order to select promising antibodies or even design new ones. Even though a variety of knowledge-primed methods (e.g. force fields) have been adopted for affinity prediction, the challenge of accurately predicting binding affinity between antibodies and antigens remains.

In this thesis, we propose a purely data-driven approach to predict antibody-antigen binding affinity using geometric deep learning methods. Our research aims to evaluate the effectiveness of graph neural networks (GNN) and compare it to a state-of-the-art force field-based method. In order to achieve this, available crystallized antibody-antigen complexes are converted to graph structures and used to train GNN-based learning methods. In addition, given the scarce availability of training data for the antibody-antigen affinity prediction problem, we explore the potential of transfer learning to improve predictive performance (e.g. through the inclusion of general PPI complexes).

The implementation of our designed GNN provides a PyTorch framework for generic binding affinity prediction using graph-like structures. The trained GNN outperforms the force field baseline on a diverse set of antibody-antigen complexes by showing robust results across low- and high-quality structures. The implemented transfer-learning techniques did not result in significant performance improvements, although numerous such techniques remain to be explored. The effectiveness of GNNs for affinity prediction and their end-to-end differentiability highlights their potential for studying the mechanisms of antibody binding. These properties further allow applications in the improvement and de novo design of experimental and therapeutic antibodies.

Contents

CHAPTER 1

Introduction

1.1 Motivation

The necessity to react to ever-changing environments and pathogens requires the ability to adapt to different situations quickly. Therefore, the family of jawed vertebrates has developed a mechanism to produce a specific class of proteins called antibodies. These proteins can bind to specific exogenous proteins (also referred to as antigens) leading to downstream immune responses or cluttering of the bound proteins, thus making them ineffective [CA06].

The functionality of antibodies can also be used therapeutically by producing antibodies in animals or cell cultures and administering them to patients. The computational design of new antibodies that bind to specific antigens has seen increasing attention in recent years [NAB+20]. Here, a major remaining challenge is to predict the *binding affinity* (binding strength) between designed antibodies and antigens.

The basic structure of antibodies and the principles guiding antibody-antigen binding have been studied extensively over the last decades [BGM+88, SFW93]. Methods have been developed to measure the location of atoms in molecules that allow a 3D representation of an antibody bound to an antigen (Figure 1.1).

The binding between an antibody and antigen is based on non-covalent and mostly weak interactions between the atoms of both proteins [SCKO13]. These interacting forces can be modeled using biophysical knowledge. The 3D representation in combination with this biophysical knowledge led to the development of force field methods. Such methods describe the energetic landscape of molecules with so-called force fields and based on the atom coordinates derive the energy present in a molecule. Although these methods are capable of modeling some aspects of protein-protein binding, the underlying physical interaction functions are not yet precise enough to accurately predict antibody-antigen binding affinity.

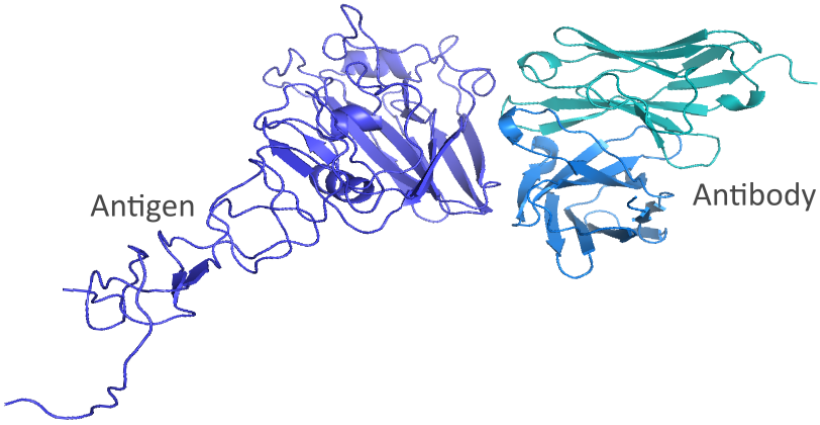


Figure 1.1: Cartoon representation of an antibody-antigen complex (PDB ID: 5W6G). Focus on the 3D structure of the amino acid chains. Only the binding site of the antibody is shown.

Recent advances in machine learning, specifically in the domain of molecular biology, show promising results for a multitude of tasks [JLGWS21, ZCH+20]. Especially deep learning-based antibody structure prediction and generation methods could benefit from a binding affinity prediction method written in the same framework to improve the structures of bound proteins.

Antibody-antigen complexes are highly diverse structures containing a lot of information leading to a high dimensional numerical representation. Deep learning frameworks have the potential to learn expressive representations of high-dimensional data in lower-dimensional space for a specific task. Especially, graph neural networks are able to exploit the inherent spatial structure of protein complexes and utilize geometric priors to extract relevant information [BBCV21]. However, graph neural networks, like most deep learning approaches, require many data points during training.

Determining the bound structure of antibody-antigen complexes and their affinity experimentally proves to be a cumbersome task leading to few available data points. This suggests the incorporation of related data by applying suitable algorithms while training the models. Different, but related, research fields (eg. drug-target interaction [ZZW+22] or deep mutational scanning data [HMW+22]) have received more attention and, therefore, more data are available. These data could possibly provide valuable information for antibody-antigen affinity prediction.

1.2 Aim of the work

In the last three decades, different binding affinity prediction algorithms have been developed, ranging from simple algorithms to machine learning and now deep learning methods [SDW+20, SZ20, VCC+18]. A major focus of this research was on the evaluation

of small molecules for drug development. The interaction between two proteins, and especially between an antibody and an antigen, has been insufficiently studied and state-of-the-art models are not sufficiently accurate [TRA+19]. Another challenge is the sparsity of training data for this specific type of interaction that hinders the development of machine learning approaches.

A promising way to improve the prediction of binding affinity is to better incorporate structural information and leverage various data and training modalities. This includes in particular the design of a deep graph neural network, aggregation of relevant data and utilization of transfer learning suitable for the available data. This leads to the following research questions:

Research Questions

RQ1: Does a geometric deep learning approach outperform the Rosetta Energy Function regarding the predictive power of antibody-antigen binding affinity?

The first research question of this thesis is focused on the comparison of geometric deep learning methods with knowledge and statistically primed methods such as force fields. Therefore, a geometric deep learning model (Graph Neural Network) designed for antibody-antigen binding affinity prediction is implemented and compared to a baseline. The currently best available method is the Rosetta Force Field (precise Rosetta Energy Function 15 [ALFJ+17]) for the task of antibody-antigen binding affinity prediction [GVZ+21].

Both approaches are compared based on three performance measures: Root-mean-squared-error, Pearson correlation, and absolute error.

RQ2: Do transfer-learning strategies (domain and/or task, parallel and/or sequential) to overcome data scarcity limitations improve the predictive power of graph neural networks for antibody-antigen binding affinity prediction?

The scarcity of antibody-antigen data with structural information and absolute binding affinities suggests incorporating related data and applying algorithms designed for utilizing such data. The goal of this thesis is to compare transfer-learning (TF) and multitask-learning (MTL) approaches, that aim to overcome those limitations, regarding their improvement of the predictive power of antibody-antigen binding affinity (Root-mean-squared error).

Therefore, data from related domains (eg. protein-protein binding) or antibody-antigen data for a different task (eg. change in binding affinity based on a mutation) should be utilized during training. The model will either be trained on the selected data/tasks in parallel or first pretrained using the related information and then finetuned on our antibody-antigen binding affinity dataset.

1.3 Thesis outline

To answer the above-stated questions, an introductory overview of antibodies, antigens and their binding is given. Additionally, the preliminaries include a recap of the graph notation used in the thesis and an overview of geometric deep learning and transfer learning. In this chapter, the necessary background information to understand the thesis is summed up.

Following the preliminaries, a section on related work will provide an overview of the fields of binding affinity prediction and protein structure prediction. Approaches from both fields will be used for transfer learning to answer RQ2. We lay an emphasis on force field methods and give a more detailed description of the baseline, the Rosetta Energy Function 15.

The next chapter will describe the available data and analyze the differences and similarities between the antibody-antigen dataset and the related datasets. Then, the methods and implementation details for the experiments are given. Here, we present an overview of the implementation and describe the most important aspects in more detail.

Finally, the experiments to answer both research questions are introduced and in the next chapter, the results are presented and reflected. The thesis concludes with an outlook on possible ways to improve the approach presented or promising alternatives.

CHAPTER 2

Preliminaries

This chapter serves as an introduction and overview of groundwork references built upon in this thesis. First, we introduce biological information and terminology relevant to this work. Then, the concept of graph theory is reviewed and the notation used in this thesis is established. Lastly, we outline the concepts of graph neural networks and transfer learning.

2.1 Antibody-Antigen Binding

The guiding principles underlying antibody-antigen binding (and in general protein-protein interactions) are not yet fully understood. This stems among other things from the fact that such interactions, as well as the involved binding partners, are found in a wide variety regarding their structure and binding sites [NT03]. A slightly distinct but highly diverse group is that of antibody-antigen interactions. To understand the uniqueness of this group, we highlight the common structure of antibodies first.

2.1.1 Antibodies

The basic composition of four polypeptide chains[1] compromising an antibody is the same across all different variants. As shown in Figure 2.1 an antibody consists of two heavy (H) chains and two light (L) chains. Both the L- and H-chains are identical and are linked by disulfide bonds. Together they form the typical Y-shaped form of antibodies with the "arms" of the structure being called the antigen-binding fragments (F_{ab}) as seen in Figure 2.1B [SCKO13].

Each F_{ab} contains two variable fragments (V_H and V_L in the H- and L-chain respectively) that together form the variable fragment (F_V) of the F_{ab}. The binding to an antigen

[1] Amino-acids linked by peptide bonds

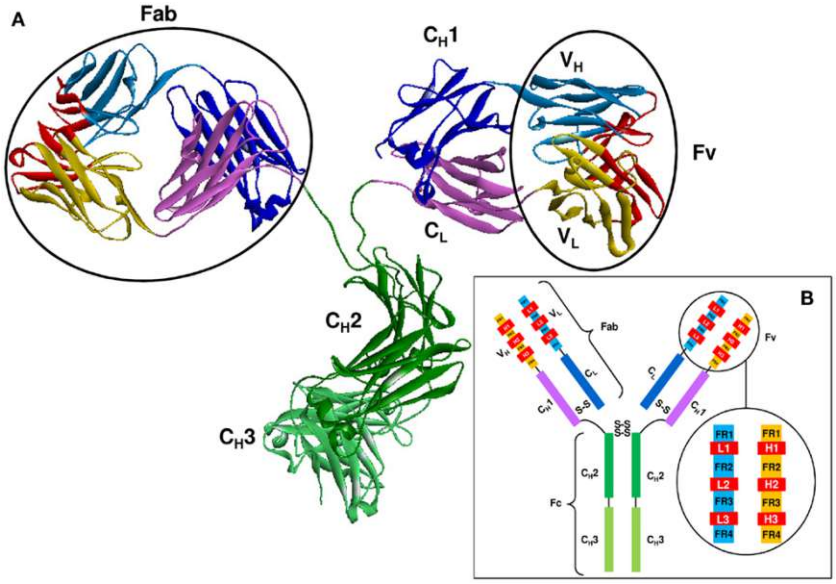


Figure 2.1: (A) Cartoon representation of the 3-D structure of an antibody molecule (PDB ID: 1IGT). (B) Schematic view of the antibody molecule [SCKO13]

happens through the F_V and is therefore called the antigen-binding-site or paratope. Each variable fragment (V_L & V_H) is composed of three hypervariable loops that play an important part in the high specificity and affinity of individual antibodies [SCKO13].

2.1.2 Binding Mechanism

Binding between antibodies and antigens occurs by non-covalent[2] forces that are not persistent and can be disrupted by a change in the environment. As seen in Figure 2.2 the counterpart of the paratope on the antibody, the binding site on the antigen, is called an epitope [Jan01].

In the context of this thesis only the F_V of the F_{ab} that actually binds an antigen is considered and shown (Figure 2.2 right side). The antigen (Figure 2.2 left side) is one of the surface proteins of the full influenza virus, that acts as a target for this antibody. Together they form a bound antibody-antigen complex, or simply complex.

The amino acids (referred to as residues within polypeptide chains) of the paratope and epitope interact in a bound complex through weak and non-covalent forces. These forces are described in detail by Janeway et al. [Jan01] and summarized below:

[2]No shared electrons in the interaction

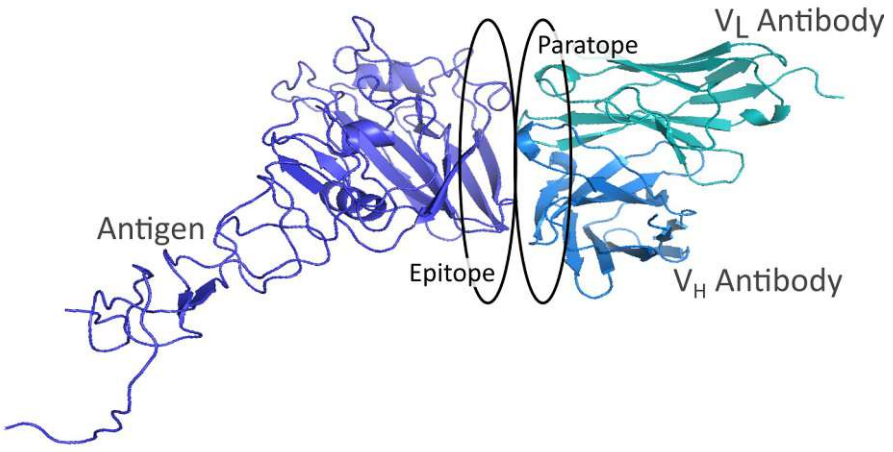


Figure 2.2: Cartoon representation of the F_V of an antibody bound to an antigen (here, an influenza virus protein) (PDB ID: 5W6G)

1. Electrostatic forces: Attraction between opposite charges
2. Hydrogen bonds: Sharing of hydrogen atoms
3. Van-der-Waals forces: Fluctuations in electronic charge induce dipoles and lead to attraction between neighboring atoms
4. Hydrophobic forces: Hydrophobic surfaces are inclined to group together to exclude H_2O molecules

The interactions mentioned above are individually rather weak forces contributing only a few calories per mole. However, together they can result in a reasonable binding energy of 12 kcal/mol, which can be found in the typical interaction of antibodies and antigens [PLJY14].

The definition of paratope and epitope is not entirely coherent in the literature, as it is often based on an arbitrarily chosen distance cutoff (given in Ångström[3]) that defines interacting residues. In our experiments, we applied a commonly used cutoff value of 5Å ([GPC+05, PLT01, LSJ08]) and all residues in the antibody or antigen that have a counterpart in the other protein are considered to be part of the paratope or epitope, respectively.

In the further context of this thesis, the paratope and epitope together are termed the binding interface (or simply interface) and residues in either epitope or paratope will be called interface residues.

[3]Ångström (Å) is a distance measure commonly used in structural biology: $1Å = 10^{-10}m$

Binding affinity measurements

There are multiple ways to measure the energy involved in the binding of two molecules. One of the most commonly employed ways for proteins is the measurements of the concentration of each binding partner (A & B), as well as of the complex AB.[4] The binding affinity determined by such experiments is usually described in terms of the dissociation equilibrium constant or K_D [Pol10].

In general, the biomolecular reaction is described as below:

$$A + B \rightleftharpoons AB \tag{2.1}$$

The right-facing arrow indicates the binding reaction, while the left-facing arrow specifies the dissociation of AB to A and B. For both reactions we can define:

$$\text{rate of binding} = k_+(A)(B)$$
$$\text{rate of dissociation} = k_-(AB)$$

with k_+ and k_- being the association and dissociation rate constant, respectively, and () indicating the concentration of the molecule [Pol10].

An equilibrium constant is, by definition, measured in an equilibrium of the binding and dissociation rates given by:

$$\text{rate of binding} = k_+(A)(B) = k_-(AB) = \text{rate of dissociation}$$

To be precise, the dissociation equilibrium constant is equal to the ratio of the binding rate and dissociation rate or the ratio of the concentration of the free molecules (A_{eq}) and (B_{eq}) and the bound complex (AB_{eq}) in equilibrium [Pol10].

$$K_D = \frac{k_-}{k_+} = \frac{(A_{eq})(B_{eq})}{(AB_{eq})} \tag{2.2}$$

Binding affinity can also be described by quantifying the physical interaction forces as the Gibbs free energy (ΔG) that is directly related to the thermodynamic K_D value by

$$\Delta G = -RT \ln \frac{1}{K_D} \tag{2.3}$$

with R being the gas constant and T the absolute temperature of the K_D experiment [Pol10].

[4] There are many difficulties and nuances to consider while measuring the binding affinity of two molecules. More information in [Pol10, JAVH20]

2.1.3 Modeling protein complexes as graphs

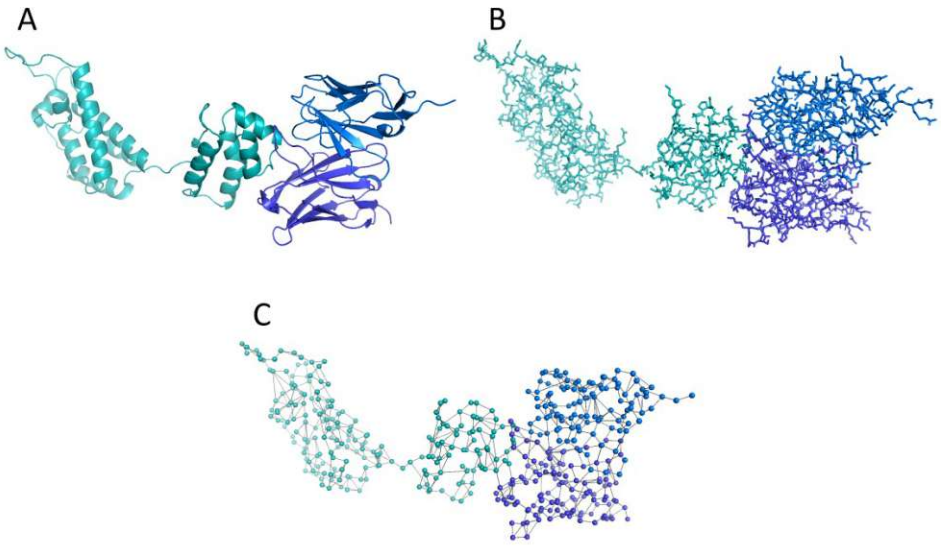


Figure 2.3: Examples for graph representations of a protein complex (PDB ID: 5W6G) A) Cartoon representation B) Sticks representation C) C-alpha atoms with 5Å-proximity edges

Protein complexes (as well as single proteins) can be represented in different forms depending on the conveyed information. Focusing on the surface of proteins we can model them as 3D objects (Volumes) or with the amino acid chain in mind they can be represented as character sequences. However, if the focus lies on the 3D-position of each residue, protein complexes can also be represented as graphs.

Figure 2.3 shows three reasons why a graph representation is suitable for protein complexes:

(A) Sub-Figure A shows the cartoon representation of proteins, which is a common representation in the fields of molecular biology and chemistry. This representation indicates the residue sequence of proteins (amino acids linked by peptide bonds). In this case, residues could be seen as nodes and peptide bonds could be seen as edges.

(B) Sub-Figure B shows the stick representation that focuses on atoms and their covalent bonds[5]. Here atoms could be interpreted as nodes and the covalent bonds as edges.

(C) Sub-Figure C shows the residues (here only the C_α atoms are shown) as spheres and the 5Å proximity edges. Therefore, a graph can be built based on the proximity of residues (or atoms) with edges if two residues are closer than a certain proximity threshold.

[5]Covalent bonds involve sharing of electrons between atoms

2.2 Graph Theory

The mathematical field of graph theory helps to formalize the above-derived graph and enables numeric representations for this kind of data structure. The scientific endeavor towards a graph theory started with the publication of Leonard Euler called *Seven bridges of Königsberg* in 1736 [BLW76]. More than 100 years later the term *graph* was introduced by Sylvester [SYL78] and over the years the problem was formalized in a way as known today.

Definition 2.2.1 (graph)**.** A **graph** $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is defined as a pair of a finite set of **nodes** $\mathcal{V} = \mathcal{V}(\mathcal{G})$ and a finite set of edges $\mathcal{E} = \mathcal{E}(\mathcal{G}) \subseteq \mathcal{V} \times \mathcal{V}$. An **edge** $e \in \mathcal{E}(\mathcal{G})$ is **directed**, therefore an ordered pair $e = \{v_1, v_2\}$ of nodes $v_1, v_1 \in \mathcal{V}(\mathcal{G})$, or **undirected** and is represented as an unordered pair $e = (v_1, v_2) = v_1, v_2$ of nodes $v_1, v_2 \in \mathcal{V}(\mathcal{G})$ [DGKP14].

The neighborhood $\mathcal{N}(n_i) = \{n_j | (n_i, n_j) \in \mathcal{E}(\mathcal{G})\}$ of a node n_i is defined as all nodes connect to n_i.

If all edges $e \in \mathcal{E}(\mathcal{G})$ are directed $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is called **directed graph**, whereas $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is called **undirected graph** if all edges $e \in \mathcal{E}(\mathcal{G})$ are undirected.

Applying this notation to the three graphs in Figure 2.3 gives the following definitions:

(A) $\mathcal{G}_\mathcal{A}(\mathcal{V}, \mathcal{E})$

$v \in \mathcal{V}(\mathcal{G}_\mathcal{A})$ for all residues v in the complex

$e = (v_1, v_2) \in \mathcal{E}(\mathcal{G}_\mathcal{A})$ if residue v_1 and residue v_2 are connected by peptide bond

(B) $\mathcal{G}_\mathcal{B}(\mathcal{V}, \mathcal{E})$

$v \in \mathcal{V}(\mathcal{G}_\mathcal{B})$ for all atoms v in the complex

$e = (v_1, v_2) \in \mathcal{E}(\mathcal{G}_\mathcal{B})$ if atom v_1 and atom v_2 are connected by a covalent bond.

(C) $\mathcal{G}_\mathcal{C}(\mathcal{V}, \mathcal{E})$

$v \in \mathcal{V}(\mathcal{G}_\mathcal{C})$ for all residues v in the complex

$e = (v_1, v_2) \in \mathcal{E}(\mathcal{G}_\mathcal{C})$ if $d(v_1, v_2) \leq x$, with

x...proximity threshold (eg. 5Å)

d...distance (eg. euclidean distance)

Numerical representation of graphs

A graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $\mathcal{V}(\mathcal{G}) = \{v_1, v_2, ..., v_n\}$ can also be represented numerically, which allows mathematical transformations. Nodes $v_i \in \mathcal{V}(\mathcal{G})$ can be represented by a vector $x_i \in R^m$ (feature vector) with m dimensions. Stacking all of these feature vectors leads to the node feature matrix.

Definition 2.2.2 (node feature matrix)**.** The **node feature matrix** $\mathcal{X}(\mathcal{G})$ of graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is defined as the combination of all the node feature vectors $\{x_i | \forall v_i \in \mathcal{V}(\mathcal{G})\}$ [DGKP14].

$$\mathcal{X} = (x_1, x_2, ..., x_n) \in R^{n \times m}$$

The edges of a graph can be represented by an adjacency matrix.

Definition 2.2.3 (adjacency matrix)**.** An **adjacency matrix** $\mathcal{A}(\mathcal{G}) = (a_{ij})$ of graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V}(\mathcal{G}) = \{v_1, v_2, ..., v_n\}$ and edge set $\mathcal{E}(\mathcal{G}) \in \mathcal{V} \times \mathcal{V}$ is a quadratic $n \times n$ matrix [DGKP14].

$$a_{ij} = \begin{cases} 1 & \text{if}(v_i, v_j) \in \mathcal{E}(\mathcal{G}) \\ 0 & \text{otherwise} \end{cases}$$

This concept can be extended to include edge information that leads to an edge tensor.

Definition 2.2.4 (edge tensor)**.** An **edge tensor** $\mathcal{E}_{\mathcal{T}}(\mathcal{G}) = (e_{ijc})$ of graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V}(\mathcal{G}) = \{v_1, v_2, ..., v_n\}$ and p-dimensional edge information $\{z_{ij1}, z_{ij2}, ..., z_{ijp}\}$ for each edge (v_i, v_j) is a $n \times n \times p$ matrix [DGKP14].

$$e_{ijc} = \begin{cases} z_{ijc} & \text{if}(v_i, v_j) \in E(G) \\ 0 & \text{otherwise} \end{cases}$$

An example of edge information z_{ijc} of edge $(v_i, v_j) \in \mathcal{E}(\mathcal{G})$ could be the Euclidean distance between nodes v_i and v_j. With this numeric representation of graphs, machine learning algorithms can be applied to learn meaningful transformations of this kind of data.

2.3 Geometric Deep Learning

Geometric Deep Learning is defined by Bronstein et al. [BBL+17] as an umbrella term for deep learning methods that generalize to non-euclidean structures such as graphs. Using this terminology they try to unify the notion of using geometric priors in deep learning systems, like convolutional filters in Convolutional Neural Networks (CNNs) and extend these ideas to other domains.[6]

A fundamental prior of geometric deep learning systems is the utilization of possible symmetries. If the transformation of an object leaves certain properties unchanged, it is said to be invariant. For example, the task of binding affinity prediction is the same irrespective of the rotation of the complex, leading to the need for a rotation-invariant function [BBCV21].

[6] More information and mathematical elaboration can be found in the publication [BBL+17] and in the pre-print of a book [BBCV21] about Geometric Deep Learning by Bronstein et al.

Definition 2.3.1. Let g be a transformation of the input x then the function f is said to be **g-invariant** if $f(g(x)) = f(x)$ [BBL+17].

If the output of a function changes in the same way as the input is transformed, the function is called equivariant to this transformation.

Definition 2.3.2. Let g be a transformation of the input x and output of function f then the f is said to be **g-equivariant** if $f(g(x)) = g(f(x))$ [BBL+17].

The task of structural binding affinity prediction is invariant to rotations and translations of the input structure leading to the need for functions that have these properties. Neural networks on graphs can be defined to enforce invariance or equivariance with respect to their input.

2.3.1 Graph Neural Networks

Due to their flexibility, graphs are employed as the primary data structure in diverse research areas including social science (social networks), natural science (protein-protein interaction), or knowledge graphs. This led to an increased focus on machine-learning approaches for graphs over the last few years.

Especially the advances in deep learning for visual computing through the use of CNNs gave rise to a great deal of interest in deep learning in general and graph deep learning in particular [WPC+21].

Graph neural networks (GNNs) can be categorized based on different criteria [WPC+21]: recurrent GNNs, convolutional GNNs, graph autoencoders, spatio-termporal GNNs. The group of convolutional GNNs will be the focus of this thesis[7].

Convolutional operations on graphs can be seen as parameterized aggregations of node information based on the graph structure (also called message passing and described in detail below). Zhou et al.[ZCH+20] display some of the similarities to convolutional filters in CNNs: Both aggregate information from a local neighborhood based on parameterized filters. In contrast to the fixed-sized neighborhood in CNNs, the local neighborhood in graphs is based on the graph topology and therefore varies from node to node. Thus, one of the major challenges of GNNs is the definition of convolutional operations that deal with differently sized neighborhoods while maintaining permutation-invariance[8].

Message Passing

The concept of graph convolutional operations is also known as message passing. The term message passing stems from the flow of information (messages) in the graph and how this process can be utilized in GNNs.

[7] More information on spatial and spectral convolutional GNNs and their difference can be found in [ZCH+20] and [BBCV21]

[8] Invariance of a function to the ordering of the input

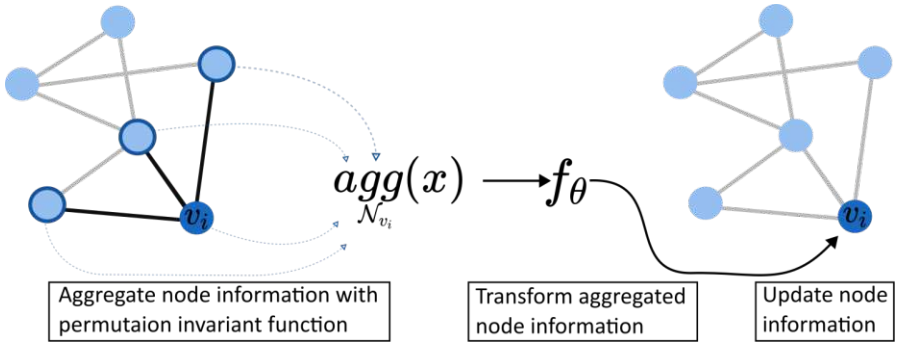


Figure 2.4: Message Passing: Three-step process of the graph convolutional operation.

Figure 2.4 shows the basic principles of graph convolutional operations. First, the local neighborhood of a node is aggregated with a permutation-invariant aggregation function (e.g. sum-, mean-, max-pooling). The aggregated information is then transformed via a parameterized function f_θ (usually a neural network) and the node information is updated. This operation is performed for every node $v_i \in \mathcal{V}(\mathcal{G})$. By stacking multiple convolutional operations, we can pass information from nodes not directly connected by an edge, and so achieve the traversal of information throughout the full graph.

As an example with local neighborhood summation as aggregation function, one of the convolutional operators used in this thesis is GCNConv [KW17].

Definition 2.3.3 (GCNConv)**.** The convolutional operator **GCNConv** on graph $\mathcal{G}$ is defined for node $v_i \in \mathcal{V}(\mathcal{G})$ with x_i being the node information for node v_i as $f_\theta(x_i) = x_i'$ with

$$\mathbf{x}_i' = \mathbf{\Theta}^\top \sum_{j \in \mathcal{N}(v_i)} \frac{a_{j,i}}{\sqrt{\hat{d}_j \hat{d}_i}} \mathbf{x}_j \tag{2.4}$$

for node-wise update with $\hat{d}_i = 1 + \sum_{j \in \mathcal{N}(i)} a_{j,i}$.

In matrix notation for the full graph

$$\mathbf{X}' = \hat{\mathbf{D}}^{-1/2} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-1/2} \mathbf{X} \mathbf{\Theta}. \tag{2.5}$$

with $\hat{\mathbf{A}}_\mathbf{T} = \mathbf{A}_\mathbf{T} + \mathbf{I}$ being the adjacency tensor with self loops and $\hat{D}_{ii} = \sum_{j=0} \hat{A}_{ij}$ being the degree matrix [KW17].

Another popular approach is to use an attention mechanism in the aggregation function to weight the information of the neighboring nodes. We used an improved version of the Graph-Attention-Convolution (GATConv) layer [VCC+18] called the GATv2Conv by Brody et al. [BAY22] in this thesis.

Definition 2.3.4 (GATv2Conv)**.** The convolutional operator of **GATv2Conv** on graphs $\mathcal{G}$ is defined for node $v_i \in \mathcal{V}(\mathcal{G})$ with x_i being the node information for this node as

$$\mathbf{x}'_i = \alpha_{i,i}\mathbf{\Theta}\mathbf{x}_i + \sum_{j \in \mathcal{N}(v_i)} \alpha_{i,j}\mathbf{\Theta}\mathbf{x}_j \tag{2.6}$$

with the attention coefficients $\alpha_{i,j}$ calculated using edge information from the edge tensor $\mathcal{E}_{\mathcal{T}}(\mathcal{G})$ and the learned parameter $\mathbf{a}$ for the computation of the attention scores:

$$\alpha_{i,j} = \frac{\exp\left(\mathbf{a}^\top \text{LeakyReLU}\left(\mathbf{\Theta}\left[\mathbf{x}_i \,\|\mathbf{x}_j\|\, \mathbf{e}_{i,j}\right]\right)\right)}{\sum\limits_{k \in \mathcal{N}(v_i)} \exp\left(\mathbf{a}^\top \text{LeakyReLU}\left(\mathbf{\Theta}\left[\mathbf{x}_i \,\|\mathbf{x}_k\|\, \mathbf{e}_{i,k}\right]\right)\right)}. \tag{2.7}$$

LeakyReLU ... Non-linear activation function [MHN13]

exp ... exponential function e^x [BAY22]

Currently, a lot of research is being done on different message passing operations and how these can be used to utilize node information and graph structure properties in an efficient and effective way. However, this thesis focuses on the applicability of GNNs for antibody-antigen binding affinity prediction. For a comprehensive review of message passing algorithms, refer to Zhou et al.[ZZW+22]. Therefore, only the GCNConv-Layer[KW17] and the GATv2Conv layer[BAY22] are considered.

2.4 Transfer Learning

Structural binding affinity prediction relies on two costly types of experiments to gather the necessary data (structure information + affinity measurements), as shown in Section 2.1.2. This leads to only a few available data points for a deep learning task and the need for approaches that allow incorporation of data from related domains. Unlike the conventional method for machine learning, which relies on the assumption that test and training data stem from the same distribution and feature space, transfer learning approaches are designed to manage scenarios with a mismatch in distribution and/or feature space.[FPRA20] This section provides an overview of transfer learning concepts and notations relevant for this thesis.

The idea of transfer learning is motivated by the knowledge that humans are able to learn more efficiently utilizing information from previous experiences. This can be information from different domains and/or tasks that is used to better perform on the challenge at hand.[FPRA20]

Definition 2.4.1 (Domain)**.** A domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$ is compromised of the feature space $\mathcal{X}$ and the marginal probability distribution $P(X)$, with $X = \{x_1, x_2, ..., x_n\} \in \mathcal{X}$ being an instance set. [FPRA20]

Definition 2.4.2 (Task)**.** A task $\mathcal{T} = \{\mathcal{Y}, f(.)\}$ consists of a label space $\mathcal{Y}$ and the objective predictive function $f(.)$. The sample data in a specific domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$ is defined as a pair $\{x_i, y_i\}$ with $x_i \in X$ and $y_i \in \mathcal{Y}$. The predictive function f should learn from the sample data to predict the label of new instances by learning the conditional distribution of instances $f(x) = P(y|x)$ [FPRA20].

Using the concepts of domains and tasks transfer learning can be defined as:

Definition 2.4.3 (Transfer Learning)**.** Transfer learning defines methods that aim to transfer information from a related source domain $\mathcal{D}_S$ and a source task $\mathcal{T}_S$ to improve the predictive function $f_T(.)$ of the target task $\mathcal{T}_T$ in the target domain $\mathcal{D}_T$ with $\mathcal{D}_T \neq \mathcal{D}_S$ and/or $\mathcal{T}_T \neq \mathcal{T}_S$ [FPRA20].

One aim of this thesis is to incorporate data from different domains with the same or related tasks during training and evaluate their impact on the performance of antibody-antigen binding affinity prediction. In the following chapter, we introduce the most relevant related works that utilize the concepts outlined in this chapter and served as the basis for our experiments.

CHAPTER 3

Related Work

This chapter introduces and discusses relevant work related to antibody-antigen binding affinity prediction using GNNs. Related work can be categorized based on either the domain and task or the methodology used. Regarding domain and task, only methods developed for binding affinity prediction on related different domains and structure prediction methods were considered. In terms of methodological-related work, the focus was on GNNs that try to capture biophysical interaction. A selection of these methods is shown in Table 3.1 and is discussed in the following sections, beginning with the available binding affinity prediction methods followed by the structure prediction methods.

Name	**Domain & Task**	**Method**	**Publication**
REF15	Domain agnostic free energy prediction	Weighted energy terms	[ALFJ+17]
CSM-AB	Antibody-antigen affinity prediction	Graph signatures + ML classifier	[MPA22]
FAST	Protein-ligand affinity prediction	3D-CNN & GNN	[JKZ+21]
OnionNet-2	Protein-ligand affinity prediction	Manual features + 3D-CNN	[WZL+21]
BindingDDG	Protein-protein relative affinity prediction	GNN	[SLY+22]
DeepRefine	Protein complex structure refinement	GNN	[MCW+22]

Table 3.1: Overview table of important related work for geometric deep learning driven antibody-antigen binding affinity prediction

3.1 Binding affinity prediction

Interaction prediction methods aim to predict if there would be an interaction between two molecules while binding affinity prediction tries to assess the strength of a given interaction. Computational prediction of protein-ligand[1] interaction, especially in small-molecule interactions, has gained increased attention over the last three decades. This stems from the hope of improved efficiency in the initial drug discovery phase. Thafar et al. categorize available methods based on their use case in interaction prediction and binding affinity prediction methods. For binding affinity prediction methods, they additionally distinguish between structure- and nonstructure-based methods, as well as between machine learning and classical methods (e.g. force fields) [TRA+19].

Nonstructure-based approaches (eg. sequence-based) will not be considered in this overview because of their fundamentally different type of input data and applied methods, but an overview is given in [TRA+19] and [ZZW+22].

3.1.1 Classical scoring functions

Using physical information for estimations of molecule forces is a common practice in the field of chemistry. Rule-based approaches in the field of molecular dynamics have reached a mature state with empirical force fields after 40 years of research and are now widely used to investigate the structure and dynamics of molecules. In chemistry, a force field is the collection of potential energy functions that can be used to derive interacting forces. Current additive protein energy functions have undergone extensive refinement and are now of a quality that allows their predictive use in pharmaceutical applications, the study of protein dynamics, and protein-protein interactions.[LGM15]

Force Fields

In general, force fields aim to describe the potential energy between atoms in a system based on experiments and calculations. The actual forces acting on certain atoms are then derived from the force field on the basis of their coordinates. In most cases, these energy terms can be categorized as bonded terms (interaction of atoms based on covalent bonds[2]) or non-bonded terms (non-covalent interaction: e.g. electrostatic forces).[Lea01]

$$E_{total} = E_{covalent} + E_{noncovalent} \tag{3.1}$$

To calculate the protein binding energy ΔE, resembling Gibbs Free Energy ΔG on a different scale, the total energy E of the bound complex and the total energies of the unbound proteins are compared. One of the most recent and best-performing force fields is the Rosetta Energy Force Field 2015[ALFJ+17]. Another well established force field is Amber[SFCW13] and these two are compared by Rubenstein et al.[RBN+18]

[1] Technical term for a molecule that binds to another (usually larger) molecule

[2] Sharing of one or more electrons

$$\Delta E = E_{total}(complex) - [E_{total}(protein_1) + E_{total}(protein_2)] \tag{3.2}$$

Rosetta Energy Force Field 2015 (REF15)

The Rosetta software suite is in development for over two decades as a collaboration of more than 60 institutaions and has strong influence on the field with the tools summarized by Leman et al.[LWL+20]. This section will give a brief introduction to the approach used in REF15 to calculate the total energy as in Equation 3.3. Here the total energy is calculated using an additive approach by weighted summation of energy terms E_i with weights w_i. [ALFJ+17]

$$E_{total} = \sum_i w_i E_i \tag{3.3}$$

Each of these energy terms is a function of geometric degrees of freedom, defined parameters Θ based on experiments, chemical identities, and amino acid information. There are energy terms defined for a variety of biophysical concepts, such as van der Waals forces, hydrogen- and disulfide bonds, or covalent bonds. A detailed description of every energy term can be found in the accompanying paper of the REF15 [ALFJ+17].

Antibody Benchmark

The antibody benchmark by Guest et al.[GVZ+21], which highlights different classical approaches to binding affinity prediction and compares their performance on a small antibody-antigen dataset, identified REF15 as the best-performing method for antibody-antigen binding affinity prediction.

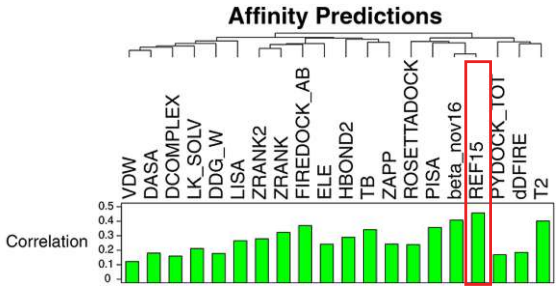


Figure 3.1: Results of the antibody benchmark adapted from Guest et al.[GVZ+21]

Guest et al. defined a small, high-quality, and non-redundant antibody-antigen dataset to evaluate docking and binding affinity approaches [GVZ+21]. A detailed introduction to the dataset is in Section 4.1. They evaluated 20 different methods, all based on energy functions derived from physical knowledge, on their dataset as shown in Figure 3.1.

3.1.2 Machine Learning approaches

This chapter structure-based machine learning methods for binding affinity prediction. The review by Thafar et al. categorizes available methods by the feature extraction type [TRA+19]. There are feature engineering-based methods (manual feature extraction),

called machine learning approaches, and representation learning methods (learned feature extraction), called deep learning approaches.

This section will first highlight an available method developed for the same task and domain as this thesis (antibody-antigen binding affinity prediction) and then provide an overview of the latest developments of deep learning-based methods for the drug-target use case.

Antibody-Antigen complexes

Currently, only a few machine learning methods designed for the antibody-antigen use case have been developed, which can likely be attributed to the low amount of data (more information in Section 4.1). To the best of my knowledge, the only peer-reviewed machine learning approach to antibody-antigen binding affinity prediction was a structure-based approach called CSM-AB [MPA22].

CSM-AB

Myung et al. describe CSM-AB as being based on modeling the interaction interfaces as graph-based signatures. These signatures are used to capture surrounding structural information and close-contact features for every atom. They consider all atoms of the interface residues and extract eight features (hydrophobic, positive, etc.) for each atom and three types of distances between the atoms based on whether they belong to the same protein or not. Atom features are aggregated for antibodies and antigens separately (counted) and distances are represented as a cumulative distribution function. In addition, external sources are used to calculate noncovalent interaction features, the distribution of residues per protein secondary structure type, and solvent-accessible surface areas [MPA22].

Finally, different supervised machine learning algorithms (Extra Trees, Gradient Boosting, Random Forest, KNeighbor, ...) are compared using these complex features by cross-validation schemes. According to the authors, the "Extra Trees" method ([GEW06]) performed best for the extracted features and was then deployed on their web server[3]. This web server was later used to gather predictions for the antibody-antigen dataset.

Protein-Ligand complexes

In contrast to the antibody-antigen task, much more data is available for protein-ligand (small molecule, eg. drug) enabling better use of structure-based deep learning approaches. The mechanism of binding for protein-ligand interactions differs from that of protein-protein interactions. Conceptually, proteins usually bind to small molecules via binding pockets, which can be imagined as small caves/holes in the protein, which neatly fit the molecule to be bound, as illustrated in Figure 3.2. Du et al. provide an overview of the different proposed binding models explaining protein-ligand binding [DLX+16].

[3]The web server can be accessed under $http://biosig.unimelb.edu.au/csm_ab/$

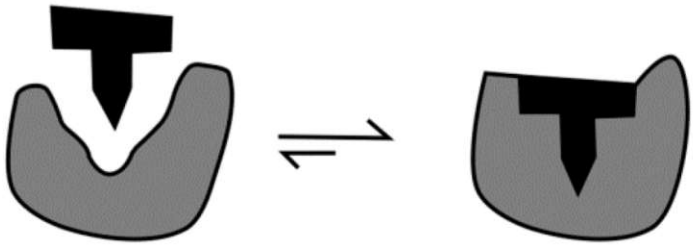


Figure 3.2: Schematic illustrations of protein-ligand binding [DLX+16] (edited)

The difference in binding mechanics leads to slightly different approaches for predicting the binding affinity. In most cases, proteins and ligands are considered separately, leading to specialized feature extraction methods for both of them. Structure-based affinity prediction either relies on manual feature extraction, solely on learned features from the raw structures, or a combination of both. As shown in the previous subsection, CSM-AB manually extracts features from the 3D complex, while this subsection highlights approaches that utilize learned feature extraction or a combination of both.

FAST

Jones et al. compared two different machine learning approaches as well as the fusion of both utilizing structural information of protein-ligand complexes. They implemented a 3D convolution neural network that works with a voxel representation of the bound complex and a graph convolutional neural network that uses a graph representation of this complex [JKZ+21].

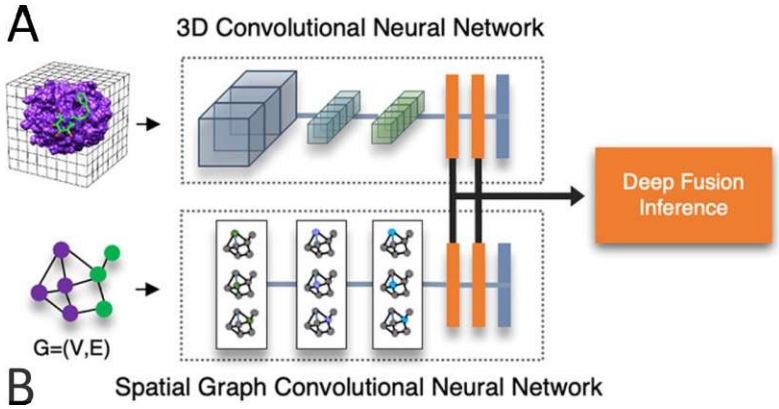


Figure 3.3: Architecture of FAST with (A) the 3D-convolutional and (B) the graph convolutional parts highlights [JKZ+21] (edited)

Sub-Figure 3.3 A shows the 3D convolutional neural network and the input representation used. The input is a $48 \times 48 \times 48 \times 19$ grid with one voxel being the size of 1Å each containing 19 atomic features. Each atom is assigned to at least one voxel based on the Van der Waals radius and feature vectors are summed if multiple atoms belong to

the same voxel. Furthermore Gaussian blur is than applied to also populate voxels not directly containing atoms to avoid sparse representations. This voxel grid is then fed into a 3D-CNN compromised of 5 convolutional layers to extract features for the full complex. These are then used to predict the binding affinity.

In Sub-Figure 3.3 B the graph approach is displayed. Each atom represents a node and edges are based on distance, grouped in covalent bond edges (distance cutoff of 1.5Å) and nonbonded edges (distance cutoff of 4.5Å). They adapt the PotentialNet architecture [FSW+18] in order to extract node features and then perform average pooling to get a graph representation used for affinity prediction.

Finally, for FAST they combine both representations to increase robustness and performance for binding affinity prediction. They compare mid- and late-fusion approaches, meaning the concatenation of intermediate layer outputs or the final representations respectively. They show that for the protein-ligand problem graph convolutional approaches outperform 3D convolutional networks and the fusion model slightly leads to a minor improvement compared to the graph convolutional model alone.

OnionNet-2

Wang et al. presented an architecture designed to combine manual feature extraction based on residue-atom distances with deep learning methods to learn a meaningful combination of these distance-based features [WZL+21].

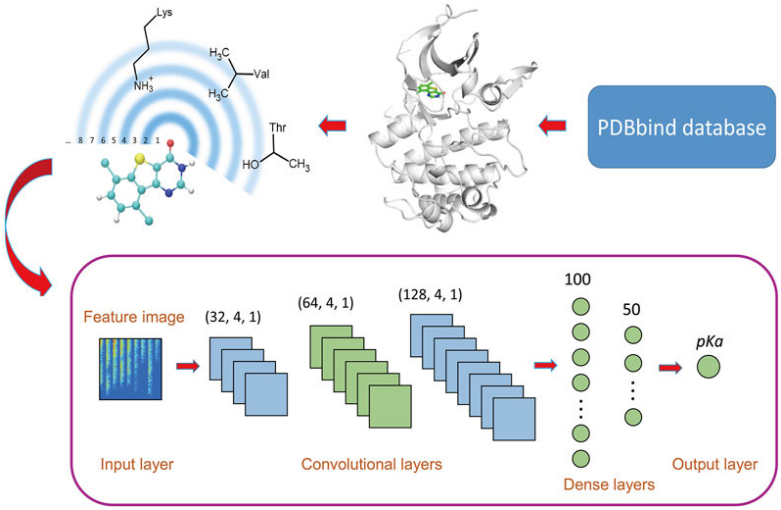


Figure 3.4: Architecture of OnionNet-2 [WZL+21]

In particular, they create feature images for every distance shell based on the count of protein-residues and ligand-atom contacts. They classified residues in 8 groups and atoms in 21 leading to $8 \times 21 = 168$ features for each distance shell, represented as a 8×21 "image".

A CNN is then used to learn representations from this distance based feature maps that distinguish between high and low binding affinity complexes. Figure 3.4 shows the full pipeline of OnionNet-2 and the left upper part displays the layered feature extraction eponymous for the name of the approach.

Relative binding affinity prediction

The task of relative binding affinity prediction is distinct, yet closely related to absolute binding affininty prediction. The objective is to predict the change in binding affinity resulting from one or multiple mutations in the paratope or epitope. The data type and the available datasets utilized for this task are further elaborated upon in Sections 4.2.2 and 4.2.3. In summary, both the 3D structures of the naturally occurring (aka. wildtype) and the mutant complex, as well as the binding affinities or the change in binding affinity are required. Several machine learning methods have been developed for this task, including the predecessor of CSM-AB, mCSM-AB [PA16], and BindingDDG [SLY+22], which utilizes GNNs. In this thesis, BindingDDG is used as a pretrained feature extractor and will be introduced in the following section.

BindingDDG

Shan et al. introduced in 2022 a GNN-based method designed for relative binding affinity prediction, called BindingDDG [SLY+22]. They build a separate graph for wildtype and mutated complex, with interface residues as nodes. Each node is connected to its 128-nearest neighbors and the relative position of the residues is used as edge weight.

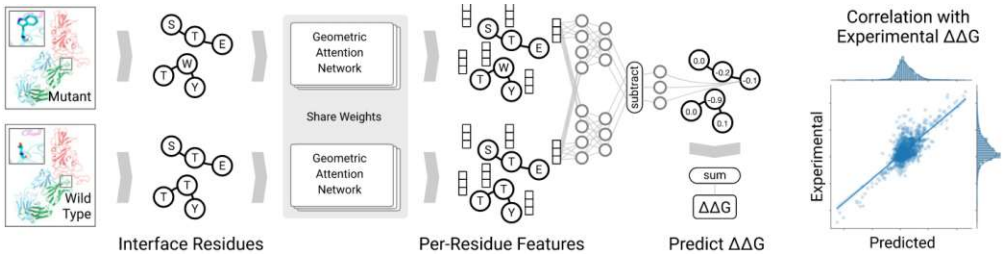


Figure 3.5: Architecture of BindingDDG [SLY+22] (edited)

They deployed an attention mechanism for message passing in their GNN to learn residue embeddings. Figure 3.5 shows the full pipeline used to predict the change in binding affinity based upon a mutation. First the same GNN is used to extract residue embeddings. Then the residue embeddings of the wildtype and mutant complex are subtracted and for each of these residue pair embeddings a multi layer perceptron is used to predict the contribution to the change in binding affinity. Finally, all these contributions are summed to get to the total change in binding affinity. Shan et al. made their code and trained model available providing an interesting feature extraction method (pretrained GNN) for the transfer learning part of this thesis.

3.2 Protein structure prediction

Modeling the structure of proteins shows some similarities with binding affinity prediction, since both operate under the assumption of learning underlying biophysical features. Structure prediction methods utilize these features for the prediction of the correct atom positions of a folded protein structure. In the past years, deep learning-based protein structure prediction has seen a number of breakthroughs, as best represented by the success of the AlphaFold2 [JEP+21] model. While AlphaFold2 primarily relies on sequence conservation information to identify interacting residues, other methods have modeled protein structures as pure graphs.

DeepRefine

DeepRefine by Morehead et al. implements a GNN to update atom positions of an already available 3D structure. Their task is to take an existing imperfect 3D structure (e.g. predicted by another model or manually designed) and correct the atom positions to get to a better, more natural, 3D structure [MCW+22].

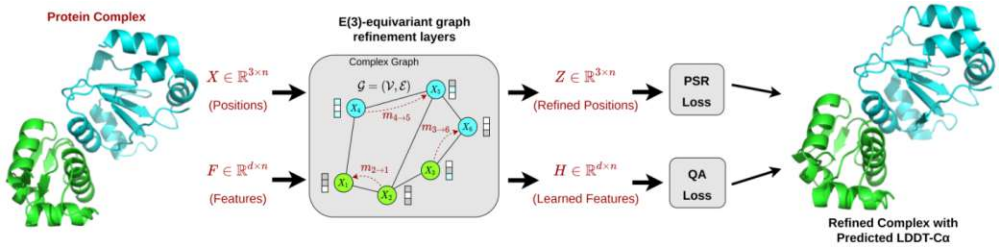


Figure 3.6: Architecture of DeepRefine [MCW+22]

The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is extracted from the initial existing structure with $\mathcal{V}$ being the atoms and $\mathcal{E}$ defined as the 20 nearest neighbors of every atom. Nodes are represented by a feature vector containing the atom type information as well as the surface proximity of this atom while edges encode information about the chain, sinusoidal edge position, relative geometric features, and information about covalent bonds. This graph serves as the input to their GNN as shown in Figure 3.6. In contrast to BindingDDG, every message passing step does not only update the node embeddings but also the positions of the nodes. This is based on the weighted distances using a learned MLP that transforms the node embeddings, edge embedding, and distance to a scalar. The predicted distances are compared to the measured ones and are used as error terms. Their objective is to learn the biophysical interaction laws between atoms and how they influence each other leading to naturally occurring structures.

The methods described in this chapter constitute the basis of the algorithms and experiments described in the following chapter. While REF15 is used as a baseline, DeepRefine and BindingDDG serve as pretrained models and the others as inspiration for feature extraction and model architecture.

CHAPTER 4

Available Data

This chapter provides an overview of the available data, how they were compiled and the similarities and differences between the transfer learning datasets and the antibody-antigen binding affinity dataset.

The application of geometric deep learning approaches to binding affinity prediction requires geometric information about the input structures as well as measured affinity values. For proteins and protein complexes, this type of data is collected through structural determination experiments (eg. X-ray crystallography[1] or NMR spectroscopy[2]) and binding affinity experiments (see Section 2.1.2 on the measurement of K_D values). Common file formats for the structural representation of molecules are PDB and mmCIF (the former is used in this thesis) and there is a common database where almost all protein and protein complexes are stored. The Protein Data Bank of the Research Collaboratory for Structural Bioinformatics (RCSB PDB) is the leading archive for those 3D structural data of biological molecules [BWF+00]. All protein complexes used for this thesis are derived from the RCSB PDB or an intermediary database and can be found using a unique identifier (PDB-ID).

For this thesis, the two most important parts of a PDB file are the header and the coordinate section. The header summarizes the protein and citation information, as well as the details of the structure determination process. The coordinate section lists all atoms, their amino acid and the 3D coordinates.

4.1 Antibody-antigen affinity dataset

Different sources of information were combined to generate our main dataset, which is referred to as the *AbAg-Affinity* dataset. This dataset is an aggregation of only non-

[1] More information in [SM00]
[2] More information in [HCH+21]

redundant antibody-antigen structures with experimentally measured binding affinity values through the combination of available resources. The following paragraphs highlight the most important aspects of the dataset generation, the partition for training and testing as well as some general characteristics.

Dataset generation

The generation of the AbAg-Affinity dataset integrates different sources adding information to the data, as shown in Figure 4.1.

The antibody structure database (AbDb, [FM18]) and the structural antibody database (SAbdAb, [DKL+14]) are curated subsets of the RCSB PDB. Furthermore, Guest et al. defined a small high-quality dataset, called the antibody benchmark (AB-benchmark, [GVZ+21]).

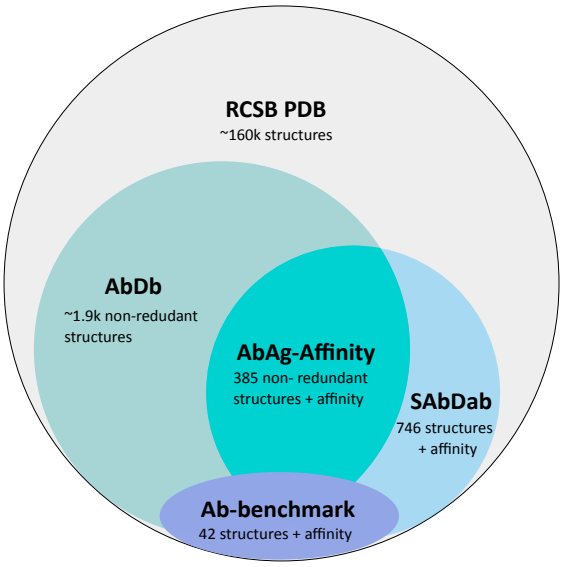


Figure 4.1: VENN diagram of datasets used for AbAg-Affinity dataset generation

AbDb provides cleaned and uniform PDB files, as well as information on redundancy between antibodies. Antibody pairs are considered redundant if all amino acids present in both antibodies are exactly the same.

SAbDab defines a subset of antibody-antigen structures with experimentally measured binding affinity values.

The antibody-benchmark (AB-benchmark) dataset incorporates only a small set of complexes (42 data points) that were selected based on stringent criteria. They defined their redundancy using the BLAST [AGM+90] algorithm with >80% sequence coverage and >98% sequence identity. Furthermore, they were also filtered according to the resolution[3] of the experiments (good resolution with $\leq$ 3.25Å resolution) and the size of both proteins (>30 amino acids).

The antibody-antigen binding affinity (AbAg-Affinity) dataset is defined as the intersection of AbDb and SAbDab while excluding complexes of the AB-benchmark as shown in Figure 4.1, leading to 385 data points. This dataset provides uniformly formatted PDB files for non-redundant complexes with measured binding affinity, excluding those values that are used in the AB-benchmark to allow comparison with other algorithms on this benchmark.

[3]Measurement of the quality of the data that has been collected

4.1.1 Benchmark

As mentioned above, the final evaluation of the trained GNN is done on the AB-benchmark dataset [GVZ+21]. This dataset consists of 42 antibody-antigen complexes with a measured binding affinity and was designed to compare different methods for binding affinity prediction.

Dataset analysis

The provided K_D values of binding affinity experiments (as described in Section 2.1.2) follow an exponential distribution and, therefore, it is common to use either the ΔG values (eg. [MPA22, SLY+22]) or their negative logarithm $-log_{10}(K_D)$ (eg. [JKZ+21, WZL+21]). These values have a well-formed distribution (roughly resembling a normal distribution) for machine learning methods, as shown in Figure 4.2. The values for $-log_{10}(K_D)$ range between 3 and 12 with an average of 8.17 and a median of 8.10. High values mark high binding affinity between antibody and antigen, while low values indicate low affinity.

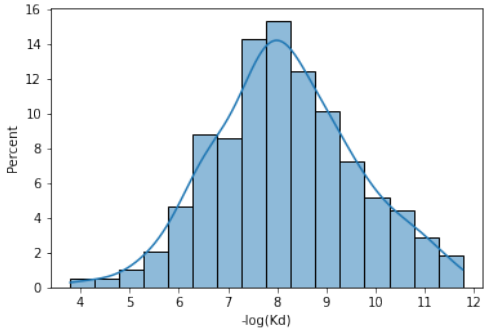


Figure 4.2: $-log(K_D)$ distribution of the full AbAg-Affinity dataset

4.2 Transfer learning data

Training deep neural networks requires many data points to learn meaningful representations of data from the target domain $\mathcal{D}_T$ and to approximate the predictive function f_T of the target task. The previous section introduced the available data in the target domain D_T of antibody-antigen complexes with the target task $\mathcal{T}_T$ of predicting the binding affinity. For the combination of $\mathcal{D}_T$ and $\mathcal{T}_T$ only a limited amount of data points are available leading to the idea of incorporating related data to improve the predictive function. This section provides an overview of the additional datasets used to evaluate transfer learning approaches for antibody-antigen binding affinity prediction.

4.2.1 PDBBind subset

The PDBBind database [WFLW04] is a subset of the RCSB PDB containing only solved structures of bound molecules. Furthermore, they also screened the primary references of these bound complexes to extract the observed binding affinity values, if available. In their latest release (2020) this database now contains more than 20.000 molecular complexes with binding affinity data [Wan20].

This dataset compromises not only protein-protein complexes but also protein-ligand, protein-nucleic acids, and other types of biomolecular complexes. The binding between proteins and ligands (small molecules like drugs - see Protein-Ligand binding in Section 3.1.2) or nucleic acids differs from antibody-antigen interactions being a subset of protein-protein interactions. Therefore, only protein-protein complexes from the PDBBind database were used. In addition, complexes were also filtered to contain only exactly two molecules that can be clearly identified. This led to 1072 protein-protein complexes with binding affinity measurements.

4.2.2 SKEMPI 2.0

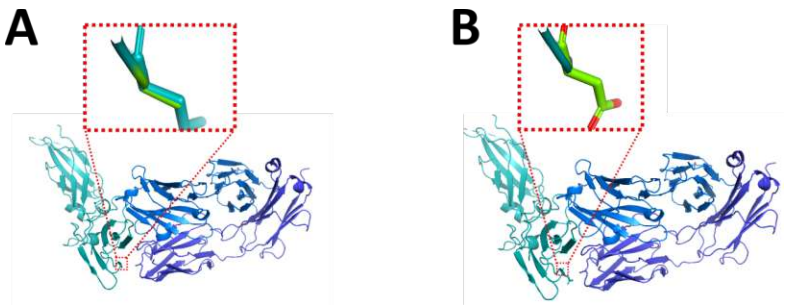


Figure 4.3: Comparison of (A) wildtype structure and (B) mutated structure from complex 1AHW

SKEMPI 2.0 is the second iteration of a database for binding free energy changes, binding kinetics and binding thermodynamics [JJGD+19]. The second version contains binding data for 7085 mutations in total. Data collected for each entry include the PDB file, the mutation as well as the affinity value of the wildtype complex and the mutated complex.

As illustrated in Figure 4.3 this database contains single point mutations (mutant only differs from wildtype in one amino acid). In this example, the mutation was on an antigen chain located at the epitope leading to a change in binding affinity.

Like the PDBBind database, this dataset also compromises different types of complexes. Again, only the subset of protein-protein complexes is selected, leading to a total number of 100 complexes. For these 100 complexes, 1629 mutations are available. The distribution of the changes in binding affinity ($\Delta - log(Kd) = -log(K_d)_{mutant} - -log(K_d)_{wiltype}$) is shown in Figure 4.4. There is a tendency of the selected mutations towards worsening

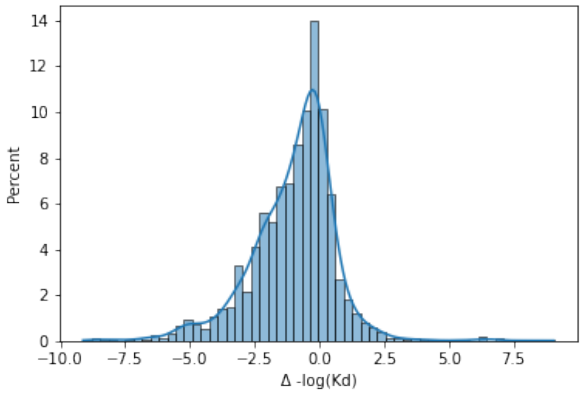


Figure 4.4: Distribution of $\Delta - log(K_d)$ of SKEMPI 2.0 subset

the binding affinity in comparison to the wildtype. However, the effect is most of the time comparatively small and close to zero.

We utilized these 1629 structures with binding affinity values as individual data points. This comes with a lot of redundancy in the dataset effectively reducing the size of the data to the 100 unique complexes.

4.2.3 Deep mutational scanning data

Deep mutational scanning (DMS) experiments can be used to measure the effects of a wide range of mutations on the binding of two molecules in a single experiment. More information on such experiments is detailed in the review of Araya and Fowler[AF11]. In general, these experiments are limited with respect to the interpretability of absolute affinity values. Nevertheless, this kind of data is available in vast quantities and may provide valuable information for antibody-antigen binding affinity prediction.

My supervisor at the Medical University (Bock Lab), Moritz Schäfer, previously curated a dataset containing antibody-antigen complexes consisting of eight DMS publications. In total 33 complexes (high redundancy between complexes within a publication) with nearly 3 million mutations were gathered. One of these publications (phillips_21_binding [PLM$^+$21]) reported actual $-log(K_D)$ values, which can be used like the other data presented above. The other authors reported control-normalized read counts, which were converted to enrichment values E, indicating binding affinity, but not directly related to K_D. These values are scaled between 0 and 1 and only provide a weak indication of the actual binding strength. Furthermore, a confidence term C_E, accompanying the E value, provides information on the precision of the measurement. Both values are used during training to sample pairs of mutations with a high likelihood of a significant difference in binding strength. An overview of the publications used to assemble the DMS dataset is provided in Table A.3.

Dataset comparison

The datasets described above carry meaningful information for the antibody-antigen binding affinity prediction task. This section will compare the related data to the AbAg-Affinity dataset and justiy their use in a transfer learning approach. Finally, an overview of all the datasets used will be given.

The distribution of $-log(K_D)$ values (Figure 4.5) shows similarities across datasets with an overrepresentation of strong binding complexes in the SKEMPI.v2 dataset and more extreme values for the PDBBind dataset. All values are in a range between a $-log(K_D)$ value of 0.6 and 15. In Figure A.2 distributions of graph descriptors are given for each dataset. All datasets show similar graph sizes in both categories (Panel A & B). The same applies to the number of edges in the interface and the full graph (Panel C & D).

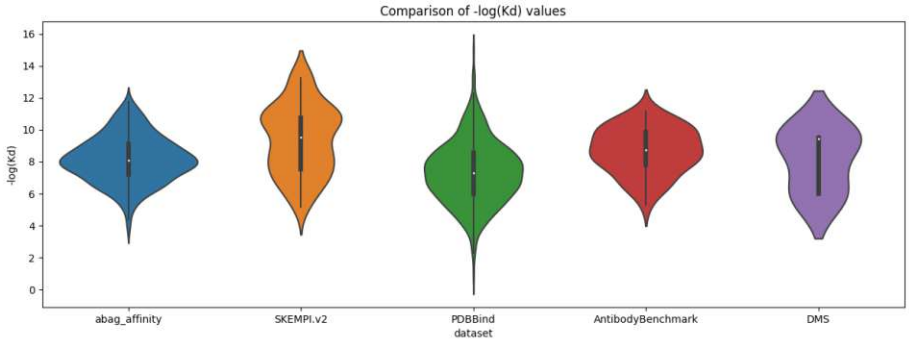


Figure 4.5: Violin plot of $-log(K_D)$ distribution for each dataset

The distribution of node types shows some differences between antibody-antigen datasets (AbAg-Affinity, AB-benchmark and SKEMPI.v2 having $> 50\%$ antibody-antigen complexes) and PDBBind. Serine and Tyrosine are slightly overrepresented in these complexes (Figure A.3).

Comparing the characteristics of the datasets summarized in Table 4.1 shows their potential for transfer-learning, but also some potential drawbacks. Our main dataset (AbAg-Affinity) shows similarities to the AB-benchmark and PDBBind dataset regarding the distribution of $-log(K_d)$ values. However, we can also observe that the generic protein-protein binding dataset (PDBBind) has on average lower binding affinity values than the antibody-antigen datasets. The relative datasets (SEKMPI.v2, DMS) have a large number of data points but only a limited amount of different complexes. This could lead to models that overfit on these complexes and do not generalize to generic binding.

Dataset	**Domain & Task**	**# complexes (# mutations)**	$\mu(\sigma)$ **-log(K_D)**
AbAg-Affinity	Antibody-Antigen absolute affinity data	446 (/)	8.21 (1.45)
AB-benchmark	Antibody-Antigen absolute affinity data	53 (/)	8.69 (1.43)
PDBBind	Protein-Protein absolute affinity data	1,072 (/)	7.35 (1.95)
SKEMPI.v2	Protein-Protein relative affinity data	100 (1,629)	8.81 (1.81)
DMS	Antibody-Antigen relative affinity data	31 (1,748,004)	7.17 (1.51) [4]

Table 4.1: Overview table of used dataset

In the next chapter, we describe how these datasets are used to train and evaluate the geometric deep learning approach as well as compare it to REF15. While the AbAg-Affinity dataset serves as the main training dataset and PDBBind, SKEMPI.v2, DMS are used for transfer-learning, the AB-benchmark dataset is only used to evaluate both approaches.

[4] Only used 199,992 data points that actually reported $-log(K_d)$

CHAPTER 5

Methodology & Implementation

This chapter will describe the main part of this thesis, the used methodology and its implementation. Table 5.1 lists the main tools used throughout the thesis project.

First, the steps taken to gather and unify all datasets will be introduced. Then a more detailed description of the graph generation process is given followed by an overview of the GNN implementation. Finally, the procedure to generate the benchmark datasets used to compare the developed method with the force field approach will be introduced.

Tool	**Description**	**Usage**	**Version**	**Citation**
Python	Programming language	Main language throughout the thesis	3.8.13	[noa22]
Snakemake	Workflow management system	Reproducible & scalable data analysis	7.8.2	[MJL+21]
Jupyter	Interactive computing software	Data exploration & analysis	4.10.0	[KRKP+16]
Weights & Bias	Platform for experiment tracking	Online logging & Hyperparameter search	0.12.16	[Bie20]
Rosetta suite	Software suite for analysing proteins	Force-Field result generation	2017.29. 59598	[ALFJ+17]
Slurm	Cluster management & job scheduling system	Running experiments on clusters	21.08.8-2	[YJG03]
PyMol	PDB visualization & tool	Visual data analysis	2.5.4	[Sch]

Table 5.1: Overview table of the core software used throughout the thesis

5.1 Data Assembly

The data collection process compromises three main steps. First, potential datasets were gathered, followed by an exploratory analysis. Then, Snakemake pipelines were implemented facilitating reproducibility of the dataset generation process as well as an easy way to integrate new data (refer to Chapter 4 for insights and visualizations).

5.1.1 Dataset search

The search for suitable data consisted of an internet search in combination with the analysis of datasets used in related works. This led to the Antibody benchmark [GVZ+21], SAbAb [DKL+14], AbAb [FM18], PDBBind [WFLW04] and SKEMPI.v2 [JJGD+19] datasets.

The DMS dataset was assembled and analyzed and the necessary structures were generated by the thesis supervisor. For this dataset, a sightly adapted pipeline was implemented only focusing on generating the mutated structures and converting the metadata file to the standardized format used throughout the thesis.

5.1.2 Exploratory analysis

To get an initial overview of the datasets, interactive Jupyter notebooks were used. In particular, the relationship between provided K_D, ΔG and temperature values was studied, as well as the possibilities to parse and manipulate PDB files using the Python programming language. PyMol was used to visualize the complex structures and deepen the understanding of the binding interface.

Summarized, this analysis showed that for the antibody-antigen use case, not a lot of data are available. The relevant datasets provided only structure and redundancy information (AbDb) or affinity values (SAbDab). This led to the combination of both datasets by taking the common complexes (based on the PDB-ID), providing 427 non-redundant complexes with affinity values. The comparison with the AB-benchmark showed some redundant complexes, that were excluded from our dataset, as described in Section 4.1.

5.1.3 Dataset generation

Based on the results of the exploratory analysis, reproducible and scalable data generation pipelines were implemented using Snakemake [MJL+21]. Snakemake workflows are implemented in a Python based language and integrate seamlessly together with Python and Bash scripts. Furthermore, Snakemake pipelines can be easily scaled to clusters and software environments can be specified for each job (a subpart of a workflow executing a script with specific input and output data). For each dataset a workflow was designed, using the information from the previous analysis, to generate a standardized metadata file and all necessary PDB files.

In general, a workflow consists of the following three steps:

1. Download data

   Using a Bash script all the data (structure + metadata) is downloaded from the respective web address. Since most of the databases are extended on a regular basis, a re-execution would likely lead to slightly different dataset sizes in the future rendering full reproducibility impossible.

2. Convert metadata & check redundancy

   The available metadata information is converted to a standardized format containing the PDB ID, the affinity value and information about the available protein chains in the complex. Chain information assigns amino acid chains to the antibody or antigen. For the AbAg-affinity dataset, the SAbDab metadata file, containing the affinity values, was used in combination with the consistent structures of the AbDb.

   In addition, a redundancy check was implemented in this part of the workflow to ensure non-redundancy between training and testing datasets. Redundancy was defined as a sequence alignment score greater than 80%. The sequence alignment score measures the overlap of characters of two differently sized sequences. Here, the amino acid sequence of each chain is compared with the sequences of the other complex using an implementation of the BioPython [CAC+09] module. Although this check is rather strict since antibodies could be different even though their sequences have a large overlap, it was done to ensure that no other dataset includes complexes redundant to those in the AB-benchmark and AbAg-affinity test datasets. For the transfer learning part, the redundancy check was used to ensure nonredundancy between the training data of the related datasets and the AbAg-Affinity validation set as well.

3. Prepare structures

   The available structures were filtered based on the metadata file and stored in the correct folders. Since the experimentally crystallized structures are often imprecise, physical force fields are used to locally adjust the atom positions[1]. Therefore, all structures were relaxed using the Rosetta Suite[2]. In our case, we used REF15 [ALFJ+17] to score the structures and the Relax protocol of the Rosetta suite to adapt the atom coordinates to minimize the total energy.

   For relative binding affinity datasets (SKEMPI.v1 & DMS), affinity information for individual mutations is present, but the structures are missing. Therefore, a closely matching structure (wildtype structure, containing no mutations) is used as a starting point to generate the mutated structures. Again the Rosetta suite was used to perform all necessary mutations and repacking. Repacking adjusts the coordinates of atoms to minimize the energy of the structure. Repacking, in contrast to relaxation, only adjusts the sidechains of the mutated residues and is therefore much faster.

[1] More information provided in [CTD+14]
[2] Documentation of the Rosetta Suite under https://www.rosettacommons.org/docs/latest/

5.2 Graph Generation

The steps of the previous section lead to datasets containing PDB files and affinity values. This section explains the subsequent conversion from these PDB files to graphs useful for deep learning. First, the necessary preprocessing steps are outlined, followed by a detailed description of the steps taken to generate the graphs used as input to the GNN. Finally, an overview of the possible parameters of the graph generation process is provided as well as the numeric representation of these graphs in the processing pipeline.

1. Preprocessing

   Although specifications for PDB files exist on the official RCSB website [Gre], they are loosely enforced, leading to a number of irregularities in some of the available files. Therefore, the first step of the pipeline is to clean the available PDB files using available tools such as *pdb-tools* [RTTB18] and *Biopandas* [Ras17]. Irregularities, that lead to errors in the subsequent processing pipeline like varied numbering schemes and untypical or missing residues/atoms, were also removed.

2. From PDB file to graph

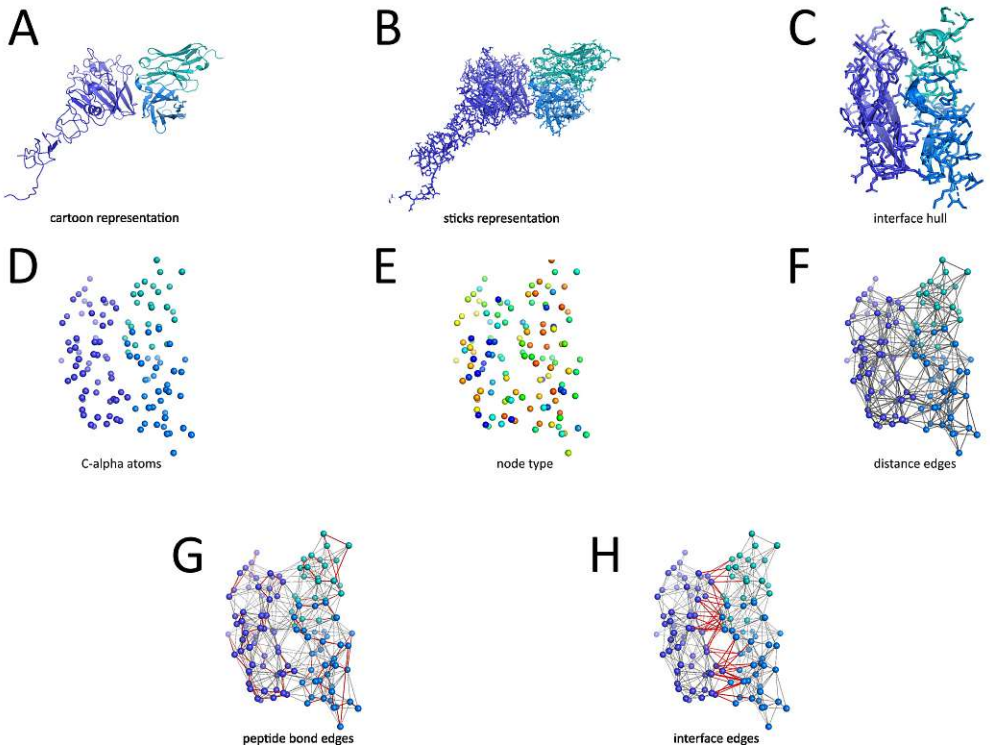


Figure 5.1: Illustration of the graph generation process: From 3D structure to graph

In Figure 5.1 the generation process from the initial structure in cartoon representation (Sub-Figure A) to a graph represented as nodes and edges (Sub-Figure H) is

shown. Sub-Figure B visualizes the complex in the sticks representation of PyMol exposing all atoms and their covalent bonds. In order to minimize the size of the graph, only the relevant parts of both binding proteins were used. The relevant part was defined as the interface atoms or residues (5Å interface) extended by a fixed distance (7Å hull size). The part of the complex was denoted as *interface hull* (Sub-Figure C). The hull size was chosen after a visual inspection of some of the complexes, taking the resulting graph size and information content into consideration.

Using only this interface hull, graph nodes are defined as either residues or atoms. While atom graphs simply use all available atoms and their coordinates as nodes, residue graphs consist only of the respective C-alpha atom of the residue (Sub-Figure D shows the residue graph). Employed node features are for example the atom or residue type (Sub-Figure E).

Edges are based on the distances of the nodes, covalent bonds or due to common node features (same chain, same protein, same residue, etc.). Sub-Figure F shows the edges of the 5Å-proximity graph, connecting all nodes with a distance smaller than 5Å. These edges also encode information used in the GNN, like distance or if the residues are connected via a peptide bond. Finally, Sub-Figure G highlights peptide bonds while Sub-Figure H colors interface edges.

3. Graph configuration

Some examples of different graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ configurations are given in the previous paragraphs. The following paragraphs will define all possible graph types and introduce the notation used for them.

Nodes $\mathcal{V}(\mathcal{G})$:

Let C be a protein-protein complex and $\mathcal{R}(C)$ be the residues and $\mathcal{A}(C)$ the atoms of this complex. Then the residue graph is defined with nodes $\mathcal{V}(\mathcal{G}_\mathcal{R}) = \{v | v \in \mathcal{R}(C)\}$ and nodes of the atom graph as $\mathcal{V}(\mathcal{G}_\mathcal{A}) = \{v | v \in \mathcal{A}(C)\}$.

The feature vector x_i of a node $v_i \in \mathcal{V}(\mathcal{G}_\mathcal{R})$ has dimension 35 and that of a node $v_i \in \mathcal{V}(\mathcal{G}_\mathcal{A})$ has dimension 69. Atom node encodings consist of the respective residue encoding and additional atom specific features. Table A.1 and Table A.2 list the features of residue and atom graphs respectively in more detail. Most of the features were implemented as previously described in [JLZ+20].

Edges $\mathcal{E}(\mathcal{G})$:

The set of edges $\mathcal{E}(\mathcal{G}) \subseteq \mathcal{V}(\mathcal{G}) \times \mathcal{V}(\mathcal{G})$, with an edge e_{ij} being the connection between node v_i and node v_j, is defined through a maximal distance cut-off value D_G. The distance of two atoms $d_A(v_i, v_j)$, with $v_i, v_j \in \mathcal{V}(\mathcal{G}_\mathcal{A})$, is the euclidean distance of their coordinates and the distance of two residues $d_R(v_i, v_j)$, with $v_i, v_j \in \mathcal{V}(\mathcal{G}_\mathcal{R})$, is the euclidean distance of their closest atoms.

Atom edges $\mathcal{E}(\mathcal{G}_\mathcal{A}) \subseteq \mathcal{V}(\mathcal{G}_\mathcal{A}) \times \mathcal{V}(\mathcal{G}_\mathcal{A})$ are defined as $\mathcal{E}(\mathcal{G}_\mathcal{A}) = \{e_{ij} | n_i, n_j \in \mathcal{V}(\mathcal{G}_\mathcal{A}) : d_A(v_i, v_j) \leq D_G\}$. The same applies to residues edges $\mathcal{E}(\mathcal{G}_\mathcal{R}) = \{e_{ij} | n_i, n_j \in \mathcal{V}(\mathcal{G}_\mathcal{R}) : d_R(v_i, v_j) \leq D_G\}$.

Each edge e_{ij} is also encoded with an edge feature vector z_{ij} with dimension 3. For atom edges $e_{ij} \in \mathcal{E}(\mathcal{G}_\mathcal{A})$ this includes the distance scaled between 0 and 1, with 1 being the D_G distance, information if both nodes belong to the same residue and information if they belong to the same protein. Residue edges $e_{ij} \in \mathcal{E}(\mathcal{G}_\mathcal{R})$ also encode the distance and the same protein, but instead of the same residue, they include information if both nodes share a peptide bond (= are neighbors on the polypeptide-chain).

Graph size:

The final size of the graph is determined using a maximum interface distance D_I and interface hull size D_H and optionally a maximum graph size S_G.

Let $\mathcal{P}_i \subseteq \mathcal{V}(\mathcal{G})$ be all nodes in the graph that belong to a different protein than node n_i. Then, the set of interface atoms $\mathcal{I}_\mathcal{A} \subseteq \mathcal{V}_\mathcal{A}(\mathcal{G})$ is defined through the interface distance D_I with $\mathcal{I}_\mathcal{A} = \{v_i | \exists v_j \in \mathcal{P}_i : d_A(v_i, v_j) \leq D_I\}$ and the set of interface residues $\mathcal{I}_\mathcal{R} \subseteq \mathcal{V}_\mathcal{R}(\mathcal{G})$ as $\mathcal{I}_\mathcal{R} = \{v_i | \exists v_j \in \mathcal{P}_i : d_R(v_i, v_j) \leq D_I\}$.

Then, the set of interface hull atoms $\mathcal{H}_\mathcal{A} \subseteq \mathcal{V}_\mathcal{A}(\mathcal{G})$ is defined through the hull size D_H with $\mathcal{H}_\mathcal{A} = \{v_i | \exists v_j \in \mathcal{I}_\mathcal{A} : d_A(v_i, v_j) \leq D_H\}$ and the set of interface hull residues $\mathcal{H}_\mathcal{R} \subseteq \mathcal{V}_\mathcal{R}(\mathcal{G})$ as $\mathcal{H}_\mathcal{R} = \{v_i | \exists v_j \in \mathcal{I}_\mathcal{H} : d_R(v_i, v_j) \leq D_H\}$.

The final atom graph $\mathcal{G}'_\mathcal{A} = (\mathcal{V}', \mathcal{E}')$ is an induced sub graph of $\mathcal{G}_\mathcal{A} = (\mathcal{V}, \mathcal{E})$ and includes only the interface hull nodes $V'(G'_A) = \{v_i | v_i \in H_A\}$. The same applies to the final residue graph $\mathcal{G}'_\mathcal{R} = (\mathcal{V}', \mathcal{E}')$ with $V'(G'_R) = \{v_i | v_i \in H_R\}$. If a maximal number of nodes restriction S_G is given then only the closest S_G nodes from I and then from H are used.

In the following course of this thesis, only the final interface hull graph will be considered and termed as $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Graph node feature matrix and adjacency tensors were stored as tensors using the geometric deep learning library PyTorch geometric [Fre] that is based on PyTorch and optimized for graphs.

5.3 Graph Neural Networks

While in Section 2.3.1 the most important general characteristics of GNNs are summarized, this section will describe the implementation and parameters of the GNN designed during this thesis. According to the classification of graph learning tasks of Zhou et al. the problem of antibody-antigen binding affinity prediction resembles a graph regression task [ZCH+20]. This regression can be seen as a classical supervised machine learning task, with PDB files as inputs and

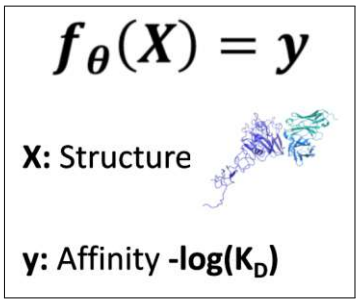


Figure 5.2: Affinity prediction as supervised ML-task

affinity values as output (Figure 5.2). Zhou et al. proposed a four-step process to design GNNs, which served as the basis for this thesis.

1. Find graph structure: Described in the previous section.
2. Specify graph type and scale: We deal with undirected, static[3] and homogeneous[4] graphs. Edges can also be seen as heterogeneous (more on that later), but are normally treated as homogeneous with a different encoding.
3. Design loss function: L2 or L1 loss function were chosen for the regression problem based on the predicted and measured $-log(K_D)$ values.
4. Build model using computational modules: In general, the GNN consists of message-passing modules (GCNConv, GATv2Conv), pooling modules and a regression head. The details are described in the following paragraphs.

The GNN designed in the last step can be seen as a processing pipeline (Figure 5.3) with each step providing different configurations with hyperparameters. The first step is the feature extraction and graph generation process that was described in the previous section. Hyperparameters for this step are the node type, edge distance, the maximal number of nodes, interface- and hull-distance, as described above.

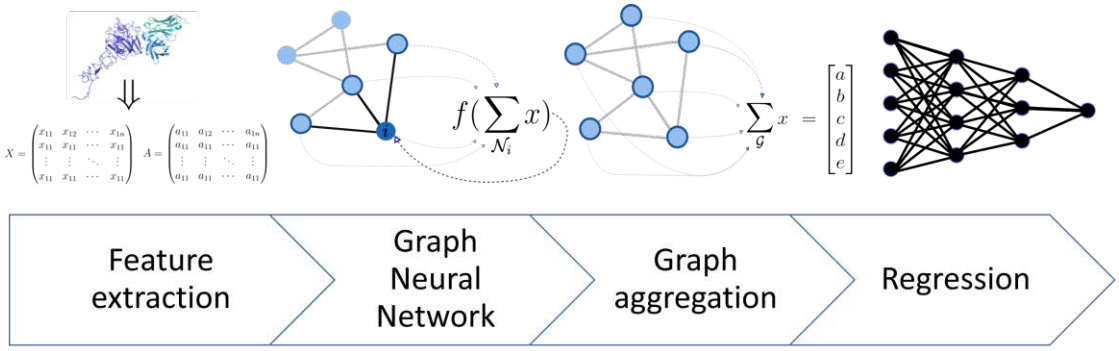


Figure 5.3: Illustration of the GNN processing pipeline: From graph structure to affinity values

The GNN step then utilized the generated graph structures and node/edge features. This step consists of message-passing layers and non-linear activation functions. The number of layers, the type of message passing layer (GCNConv or GATv2Conv) and the activation function (ReLu, GeLU or LeakyReLu) can be altered. The number of attention heads for GATv2Conv layers is also configurable. Additionally, the size of the

[3] Input features and topology of the graphs stay consistent over time [ZCH+20]
[4] All nodes have the same time (atom or residue) [ZCH+20]

hidden embeddings could be halved, doubled or left the same after every layer. Only the GATv2Conv layer is able to utilize multi-dimensional edge encodings while GCNConv only uses the node distance ($a_{i,j}$ in Definition 2.3.3 of GCNConv). Figure 5.3 shows the GCNConv with summing as the neighborhood aggregation function. The goal of modeling the physical interactions as closely as possible led to the idea of implementing a "guided" GNN. In contrast to the naive GNN, which simply uses all edges at every step, the guided GNN implements a two-step process of node information aggregation. First, nodes with a tight bond (atoms belonging to the same residue or residues having a peptide bond) are aggregated. Then, all nodes belonging to the same protein are used for message passing. This is achieved by filtering the edges based on the respective edge feature and applying the message-passing function only with the selected edges.

Following the previous GNN step, the calculated node embeddings need to be aggregated to obtain a graph embedding used in the final regression step. The implemented aggregation options include max-, mean-, sum- and attention-pooling. Figure 5.3 shows for example a sum-pooling operation. If the input graph had a fixed size (maximal number of nodes is given) then also a simple concatenation of all node embeddings would be possible with the loss of permutation invariance (nodes are sorted by interface distance).

The final step is the actual regression using the graph embeddings. Here a multilayer perceptron (MLP) was used, defined through the number of layers and a size-halving hyperparameter. Size-halving is a boolean indicator that defines if the embedding size is halved after each layer of the MLP in order to reduce the embedding dimensionality more smoothly towards 1. The output of this step is a single value (predicted $-log(K_D)$).

In an attempt to model the actual interaction in the interface, "edge-pooling" was implemented, combining the aggregation and regression steps. The idea is to predict the binding strength of each edge with a MLP by concatenating the embeddings of the incident nodes. This is done for all interface edges and the resulting edge binding strengths were summed. An overview of all available model hyperparameters and their options is given in Table B.1.

5.4 Transfer learning

As described in the previous section and chapter 4, the limited amount of available antibody-antigen data implies the utilization of transfer learning approaches to improve the predictive power of the above-described model. This section will outline the details of the two used approaches for transfer learning, reusing pretrained models and training with related data.

5.4.1 Pretrained models

Both pretrained models used in this thesis are described in Section 3, particularly BindingDDG in Subsection 3.1.2 and DeepRefine in Subsection 3.2. These models were

chosen because of their similarity to the approach described above of modeling protein complexes as graphs and utilizing GNNs to extract node embeddings.

BindingDDG is a pretrained model to embed residues based on a GNN. Their GNN is optimized to recognize the effect of one or multiple mutations on the binding affinity of a complex. The information of this model could also be leveraged to predict the absolute binding affinity, leading to the integration of this pretrained model in the pipeline and finetuning[5] it with the AbAg-Affinity dataset.

DeepRefine also utilizes a GNN to get atom embeddings and these embeddings are used to model the interaction between each atom. This interaction information is used to predict whether the atoms attract or repel each other and is utilized to update node coordinates. However, it could also offer valuable information for absolute binding prediction.

For both models, their implemented graph generation pipeline (denoted as "Feature extraction" in Figure 5.3) is used for the available AbAg-Affinity data. The pretrained models are deployed before the GNN step, optionally also excluding the GNN step and only performing graph aggregation and regression to get the absolute binding affinity predictions (compare "Graph Neural Network" in Figure 5.3).

5.4.2 Training on related data

The second transfer learning approach was training the implemented GNN directly on related data introduced in Section 4.2. In general, two methods to integrate related data during model training were implemented: *pretraining-finetuning* and *bucket-train*.

Pretraining-finetuning allows the model to first learn the distribution and peculiarities of the related dataset and then adapt the pretrained model to the target antibody-antigen data (Figure 5.4A with PDBBind as the related data). This is a common way of implementing transfer learning and follows the same intuition as using pretrained models, with the only difference being the possibility to use the complete pipeline described in this chapter.

The approach termed bucket-train (Sub-Figure 5.4B) resembles the multi-task learning idea described in [Car93]. The idea is to learn from multiple data buckets at the same time while sharing the model parameters. This should expose the model to significantly more data while still focusing on the target (here antibody-antigen absolute binding affinity) dataset while training. This approach is implemented through a custom data loader that samples data from the related dataset as well as the target dataset. In order to give the target dataset more relevance, each training epoch includes only a subset of the related datasets. This could either be the same size as the target dataset (min-sampling), laying the focus on the target dataset while not fully utilizing the related datasets, or based on the geometric mean of the dataset sized (geometric-mean-sampling), allowing an overrepresentation of related data.

[5] Optimize an already trained model on a different dataset

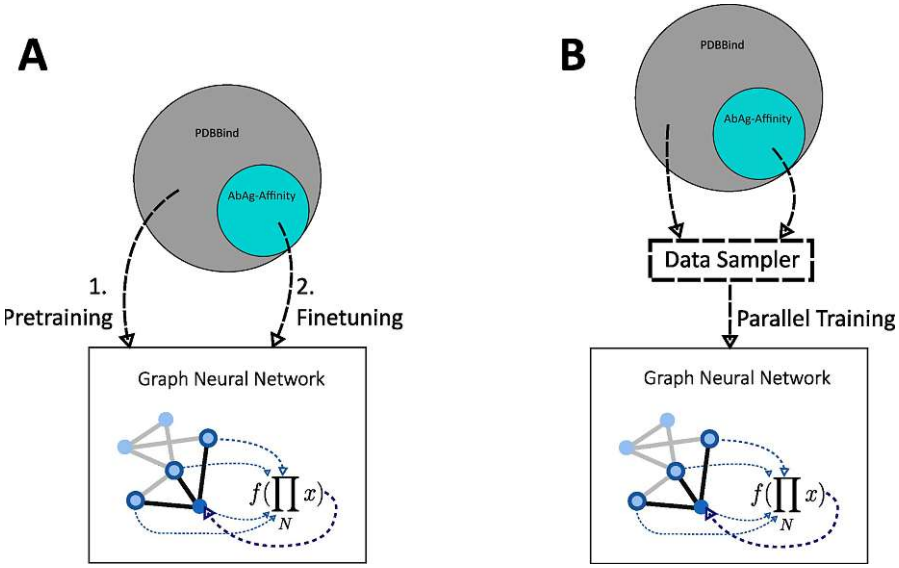


Figure 5.4: Illustration of both transfer learning approaches. A) Pretraining-finetuning method. B) Bucket-train method

The available transfer learning data can be divided into two groups. One group provides measured K_D values while samples of the second group are labeled with an affinity indicator not directly related to K_D, denoted as enrichment values E.

Data with K_D labels

K_D measurements are available for structures of the PDBBind dataset, SKEMPI.v2 and the phillips_21 publication of the DMS dataset. This data can be used in the same way as the AbAg-Affinity dataset and be easily integrated with the pretraining-finetuning or bucket-train approach. Therefore, a redundancy check is integrated to only train on data non-redundant to the AbAg-Affinity test and validation subset and AB-benchmark dataset.

The SKEMPI.v2 dataset and phillips_21 publication of the DMS dataset contain mutational data (see Table 4.1) and can therefore be used in a relative was in addition to the training on absolute K_D values. This is achieved by selecting two mutations from the same complexes and predicting their affinity values. The error term during training is then computed by the the measured difference in $-log(K_D)$ values and the predicted difference in $-log(K_D)$ values. Therefore these two datasets are used as absolute and relative datasets in the experiments described in Chapter 6.

Data with E labels

For all complexes in the DMS dataset, except phillips_21, only enrichment E values and an indicator of their accuracy denoted as NLL are provided. NLL values do not resemble

the mathematical concept of negative log likelihood but rather describe the expected precision of the E value. In order to utilize this kind of data, the affinity prediction task was slightly adapted. A binary classification to determine whether one mutation leads to a much higher binding energy than the other one was used. Hence, the first step was to identify the set of suitable mutation pairs $\mathcal{M}_s$, that were defined as having a higher difference of their E values than their average NLL values.

Definition 5.4.1 (Suitable mutation pairs)**.** Let $\mathcal{M}_a$ be all available mutations of a complex and one mutation $m_i \in M_a$ have an enrichment value e_i and NLL value nnl_i, then the set of suitable (unordered) mutation pairs $\mathcal{M}_s$ is defined as

$$\mathcal{M}_s = \{(m_i, m_j) | m_i, m_j \in \mathcal{M}_a : |e_i - e_j \leq \frac{nll_i + nll_j}{2}\}$$

For each complex of a suitable pair, the $-log(K_D)$ is predicted using the GNN and their difference is used to predict whether the first or the second complex has a higher affinity. An additional loss is integrated to scale the values to reasonable $-log(K_D)$ values by penalizing very large or very low predictions.

5.5 Benchmark

Finally, the models trained using the approaches described above are evaluated on two different datasets to answer the research questions. For one part, the independent AB-benchmark dataset [GVZ+21] will be used to compare the implemented GNN and REF15, as well as the transfer learning approaches. Since this dataset only contains high-quality data (high resolution and comparatively high binding affinity), our own more diverse dataset (AbAg-Affinity dataset) will be used in addition. The next chapter provides a detailed description of the experiments and metrics used to assess the performance on these two datasets.

CHAPTER 6

Experiments

To answer the research questions (see Section 1.2) two experiments were designed. Regarding the first question, the focus is on the performance of the GNN trained solely on antibody-antigen data and the comparison with REF15. Then, a comparison of available pretrained models and related datasets and their impact on the GNN performance on antibody-antigen data will be evaluated. This chapter will explain these experiments and how they utilize the implemented GNN, pretrained models and datasets described in the previous chapters.

Three metrics will be used to assess the model performance: Absolute Error, Root Mean Squared Error (RMSE) and Pearson correlation coefficient (Pearson's R). These metrics are often used in regression problems and Pearson's R is also prominent in publications about binding affinity prediction (introduced in Section 3) and in comparative studies [TRA+19, GVZ+21].

Definition 6.0.1 (Absolute Error)**.** The absolute error is defined as the absolute difference between a predicted value $\hat{y}$ and the measured value (label) y [Hod22].

$$e(\hat{y}, y) = |\hat{y} - y| \tag{6.1}$$

Definition 6.0.2 (RMSE)**.** The root mean squared error is defined as the square root of the mean of all squared errors of two arrays (x, y) of dimension n [Hod22].

$$RMSE(x, y) = \sqrt{MSE(x, y)} = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(x_i - y_i)^2} \tag{6.2}$$

Definition 6.0.3 (Pearson's R)**.** Pearson's R can be used to describe the linear correlation of two variables (X, Y) by dividing their covariance by the product of their standard deviations [FPP07].

$$Pearson'sR = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{6.3}$$

A set of n samples (x, y) can be used to estimate the covariance and variance, giving us the sample correlation coefficient (represented as r).

$$r = \frac{\sum_{i=0}^{n}(x_i - \bar{x}) * (y - \bar{y})}{\sqrt{\sum_{i=0}^{n}(x_i - \bar{x})^2 \sum_{i=0}^{n}(y_i - \bar{y})^2}} \tag{6.4}$$

with $\bar{x}$ and $\bar{y}$ being the mean of all samples in x and y respectively [FPP07].

In order to assess whether differences in the reported metrics are significant, the Wilcoxon signed-rank test [Wil45] will be used.

Definition 6.0.4 (Wilcoxon signed-rank test)**.** The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used to compare the locations of two populations using paired samples.

$$\begin{aligned} W &= \sum_{i=1}^{N_r} [\operatorname{sgn}(x_{2,i} - x_{1,i}) \cdot R_i] \\ W &= \text{test statistic} \\ N_r &= \text{sample size, excluding pairs where } x_1 = x_2 \\ \operatorname{sgn} &= \text{sign function} \\ x_{1,i}, x_{2,i} &= \text{corresponding ranked pairs from two distributions} \\ R_i &= \text{absolute difference based rank i} \end{aligned} \tag{6.5}$$

A non-parametric test is used because we cannot make any assumptions on the distribution of the metrics apriori as advised in [Dem06].

6.1 Graph neural networks based affinity prediction

This section will introduce the experiment used to evaluate the predictive power of GNNs for the antibody-antigen binding affinity problem and compare it to REF15. The interface size parameter was fixed at 5Å, as described in Subsection 2.1.2, and the interface hull size at 7Å (see Section 5.2). First, the remaining GNN hyperparameters are optimized and the best-found configuration is used subsequently to predict $-log(K_D)$ values of the AB-benchmark. Furthermore, a 10-fold cross-validation (CV) scheme on the full AbAg-Affinity dataset is used to get predictions for each data point as well as the RSME and Pearson's R for the 10 distinct validation subsets. This approach is used to get unbiased predictions for the full AbAg-Affinity dataset as well as a distribution of the metrics used to evaluate and compare both approaches.

Experiment design

The designed experiment is a two-step process to find the best model configuration and then compare it to REF15.

1. Exploratory hyperparameter search

   The implemented GNN is defined through multiple parameters (aka. hyperparameters) as described in Section 5.3. Initially, a random search in this hyperparameter space was done with Weights&Biases [Bie20], testing different combinations of all parameter values (see first two columns of Table B.1 for parameter ranges).

   The results of this hyperparameter search were used to gather information on the importance of each parameter and if there is a significant difference between the values of the hyperparameter. The Kruskal-Wallis test [KW52] was used to calculate if there is a significant difference in the performance of one of the possible values of a hyperparameter. This test is a non-parametric method to test whether two or more independent samples originate from the same distribution. If a significant difference was found, a post hoc pairwise test between all values was used to identify the best-performing value. Here, Dunn's test [Dun64] was used to compare the mean rank sums between two values.

2. Comparison of GNN and REF15

   Using the best model configuration found during the hyperparameter search, the GNN was used to predict the $-log(K_D)$ values for the AB-Benchmark[1]. Furthermore, the full AbAg-Affinity dataset was split randomly into 10 subsets and the model was evaluated on each of these subsets when trained on the remaining nine.

   REF15 predictions were calculated as described in Sub-Section 3.1.1 using a script provided in [GVZ+21]. To account for the different scales of REF15 predictions, the predicted energy terms were scaled using min-max scaling based on the labeled ΔG values and then converted to $-log(K_D)$ using Equation 2.3. For min-max scaling, outliers ($>$ 1 standard deviation apart from the mean) are removed from the REF15 predictions to ensure that they do not affect the determination of the min and max values. Here, the same data samples used in the GNN training were taken to calculate the parameter for min-max scaling.

   For the prediction of both methods, the absolute error terms are calculated and compared using the Wilcoxon signed-rank test, individually for the AB-benchmark and the AbAg-Affinity dataset. Additionally, the RSME and Pearson's R values for each of the validation subsets from the 10-fold cross-validation are compared. Finally, the absolute errors of the GNN for each CV-split are compared against each other to evaluate the robustness of the model.

[1] The model evaluated on the first validation split and trained on the remaining 9 was used

6.2 Transfer learning

After an initial configuration for the antibody-antigen use case was determined, different transfer learning strategies are compared. The methods are evaluated individually and then combined and compared to the GNN trained solely on the AbAg-Affinity dataset. The two backbone models introduced above (BindingDDG and DeepRefine - Section 5.4.1) and the three datasets (PDBBind, SKMEPI.v2, DMS - Section 4.2) are utilized in this experiment.

Experiment design

First, a hyperparameter search is used to evaluate the different pretrained models and related datasets for their impact on the model performance. Then, a combination of good strategies is evaluated on the same cross-validation scheme as described in the previous section.

1. Hyperparameter search

   The hyperparameter search allows the possibility of different model configurations for each pretrained model. The type of pretrained model (DeepRefine, BindingDDG, No-model) is another hyperparameter added to the configuration. Finally, only the pretrained model parameter will be evaluated and the Kruskal test will be performed to see if there is any significant difference between the three strategies.

   Related datasets will be evaluated in a similar way. Therefore, two parameters are added to the configuration: transfer-learning dataset and training strategy (compare first two columns of Table C.2). The transfer learning dataset parameter defines the dataset used in addition to the AbAg-Affinity dataset while training. The training strategy defines how these datasets are used, as shown in Figure 5.4. In this initial step, all datasets will be used individually and evaluated if there is any improvement to using no related dataset during training.

   The DMS dataset is split into 8 different subsets based on the publication. The datasets with mutations and absolute $-log(K_D)$ values (DMS-phillips_binding_21, SKEMPI.v2) are used both as absolute and relative datasets (predicting the absolute binding affinity and predicting the change in binding affinity respectively). This results in 12 different datasets that are evaluated in this step.

2. Benchmark combination of strategies

   If there are any good performing configurations, a combination of a pretrained model and related datasets is then evaluated on the AbAg-Affinity dataset and AB-benchmark dataset. This transfer learning configuration is compared to the GNN trained solely on the AbAg-Affinity data, using the same comparison as described above (Section 6.1).

All experiments are logged with Weights&Biases and the hyperparameter search is performed using Weights&Bias sweeps. The final evaluation of the hyperparameter search results, the comparison of model predictions and the visualizations are performed using Jupyter notebooks.

CHAPTER 7

Results & Discussion

In this chapter, we present the results from the above-described experiments. First, we analyze the performance of GNNs trained solely on the AbAg-Affinity dataset and compare them to REF15. Then, we discuss the results of the different transfer learning approaches. All training runs were executed on the Scientific Cluster of the Medical University Vienna (MUW) providing multiple compute nodes with NVIDIA A100 GPUs.

7.1 GNN based affinity prediction

As described in the previous chapter, a random search of the hyperparameter space was performed and the results for different configurations were compared. The best-found model is used in a comparison with REF15 on the full AbAg-Affinity dataset and the AB-Benchmark.

7.1.1 Exploratory Hyperparameter Search

A performance overview of the 228 executed training runs is given in Figure B.1. Visual inspection implies that most hyperparameters have no significant impact on model performance and lead to similar results.

After significance testing with the Kruskal-Wallis test, three hyperparameters showed a significant difference in the performance for their possible values (compare Table B.1 for a full list of the parameter values and the Krusal scores). Edge pooling leads to significantly worse results than the other methods for graph aggregation (see Figure 7.1). Mean pooling is also significantly different from max pooling.

The L1 and L2 loss functions also lead to differently distributed results but with no clear indication of which loss leads to better results. However, the L1 loss shows a lower variance in the results. The same applies to the number of fully connected layers in the

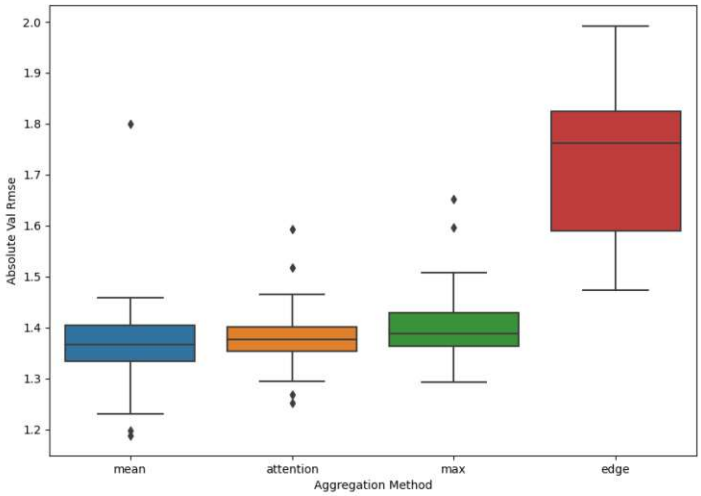


Figure 7.1: Box-plot showing the results of the different aggregation methods (Only $RSME \leq 2$ shown)

regression head. Here, using 5 layers leads to lower variance, but is not guaranteed to get the best results.

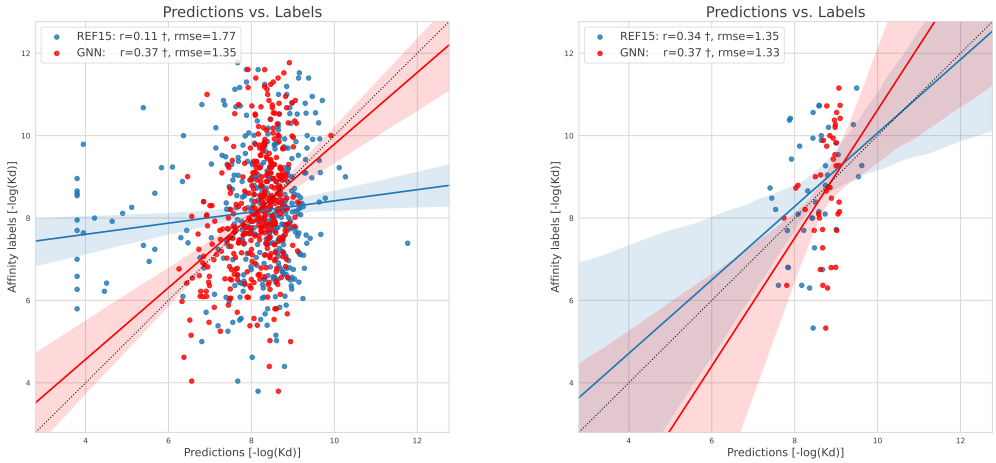
7.1.2 Comparison of GNN and REF15

The best configuration from the exploratory hyperparameter search (see Table C.1) was used for the comparison with REF15. The performance measured in RMSE and Pearson's R on the validation sets of the 10-fold cross-validation shows significant differences for GNN and REF15 (Table 7.1). Both metrics indicate that the GNN outperforms REF15 on the AbAg-Affinity dataset (lower RSME, higher Pearson's correlation). The results on the AB-benchmark show no clear difference between both approaches as REF15 performed better regarding RSME but the GNN with respect to Pearson's correlation.

	AbAg-Affinity (10-fold CV)		AB-Benchmark	
	RMSE	Pearson's R	RMSE	Pearson's R
REF15	3.11 (±1.02)	0.12 (±0.13)	1.35	0.34
GNN	**1.80 (±0.60)†**	**0.35 (±0.13)†**	**1.33**	**0.37**

Table 7.1: RMSE and Pearson's R (mean & standard deviation) for AbAg-Affinity CV). †: significant differnce

Both GNN and REF15 show a significant correlation between the predicted $-log(K_D)$ values and the labels for the AbAg-Affinity dataset (Figure 7.2a) and the AB-benchmark (Figure 7.2b)



(a) AbAg-Affinity results of GNN and REF15 (b) AB-benchmark results of GNN and REF15

Figure 7.2: Predictions vs. Labels for AB-benchmark and AbAg-Affinity dataset. †: significant correlation

The distribution of absolute error terms on the AbAg-Affinity dataset shows a superiority of the GNN compared with REF15 error terms (Figure C.1 - the majority of errors are larger for REF15). Here the Wilcoxon signed-rank test indicates a significant difference in the performances. As shown in Figure 7.2b there is no such plain difference for the AB-benchmark, which is confirmed by the Wilcoxon signed-rank test not indicating a significant difference in the error terms of both models.

While the GNN showed similar results across the AbAg-Affinity dataset and the AB-benchmark, REF15 performed better on the selected values of [GVZ+21] of the AB-benchmark in terms of Pearson's correlation (r) and RMSE. Furthermore, the performance of the GNN across the 10-fold CV shows only slight differences between the different validation sets (Figure C.2) and comparably stable predictions for the benchmark dataset with different training datasets (Figure C.3).

7.1.3 Discussion

The initial hyperparameter search did not lead to a unique set of good-performing hyperparameters but showed some interdependencies between the values of the hyperparameters. A full analysis and comparison of different configurations and their impact on the performance goes beyond the scope of this thesis and is therefore not described in more detail. For the sake of this thesis, it was sufficient to use the best-found configuration for the subsequent tasks.

Although the results of the previous section show that the implemented GNN outperforms REF15 regarding all three metrics (Pearson's R, RMSE, Absolute error) on the AbAg-

Affinity data, the results using the AB-benchmark did not provide the same conclusion (compare Table 7.1. Yet, it can be argued that this dataset does not cover the full spectrum of antibody-antigen complexes nearly as well as the AbAg-Affinity dataset (42 vs. 387 examples and rigorous manual selection of complexes for the AB-benchmark). Furthermore, the GNN provides stable results across AbAg-Affinity dataset and AB-benchmark while the results for REF15 differ a lot between AbAg-Affinity dataset and AB-benchmark (Figure 7.2). Thus, it can be argued that the GNN shows more robust results independent of the available data quality.

In conclusion to RQ1, the GNN outperforms REF15 regarding the predictive power for the antibody-antigen binding affinity problem in almost every tested scenario.

7.2 Transfer Learning

In this section, the impact of an additional pretrained node-embedding model and/or the integration of related data during training (see Section 6.2) are described and subsequentially discussed.

7.2.1 Pretrained Models

The addition of BindingDDG and DeepRefine to the GNN node embedding part of the pipeline does not lead to significant improvements. A random search in the hyperparameter space led to 107 training runs (42 with BindingDDG, 31 with DeepRefine and 34 with no pretrained model). The median performance of all three methods is very similar (Figure C.7) with BindingDDG having slightly better performance than DeepRefine. The Kruskal H-Score of 1.99 (p-value: 0.37) does not indicate that one of these approaches has a significantly different performance (RMSE) on the AbAg-Affinity validation set.

pretrained_model	# runs	median	mean
Binding_DDG	35	**1.38**	1.44 (±0.13)
No-Model	31	1.39	**1.42 (±0.12)**
DeepRefine	24	1.39	1.44 (±0.14)

Table 7.2: Hyperparameter search results for pretrained models

7.2.2 Related Data

Due to longer training times when using additional data (especially for relative training with many pairs), the number of runs for the evaluation of related datasets is limited. In total 44 training runs were successfully executed with 1-6 runs per transfer learning dataset. After removing outliers (runs with validation RSME > 2), the performance of the GNN improves when using some of the datasets (compare Figure 7.3 and Table C.3). The datasets are selected if the median performance of the respective runs (central bar in Figure 7.3) is better than the median performance of the GNN without transfer-learning

(grey background in Figure 7.3). Therefore, 8 of the related datasets are combined in the final transfer learning dataset (dataset left of the red vertical line in Figure 7.3).

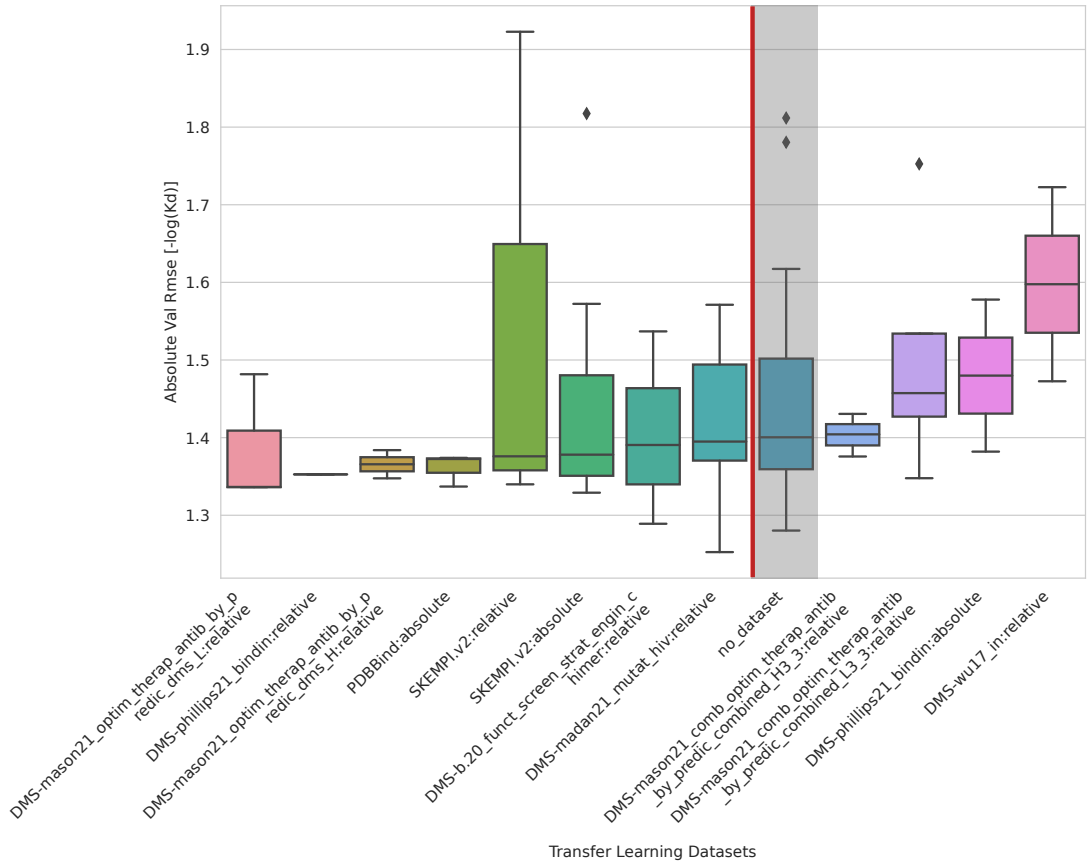


Figure 7.3: Performance of adding related datasets while training and AbAg-Affinity dataset alone using random hyperparameter configurations

Pretrain-Finetuning and bucket training

Both transfer learning methods, if used with only one of the related datasets, do not differ significantly from the use of no transfer learning at all (H-score: 1.54, p-value: 0.46). However, bucket training has slightly better mean and median performance than the other two methods after removing outliers. Bucket learning has a clearly shorter runtime than pretraining-finetuning (mean of 22.6 min and 897.8 min respectively). Therefore, for the final evaluation of the transfer learning approach, bucket learning will be selected as the transfer learning strategy.

train_strategy	# runs	median	mean
bucket_train	19	**1.38**	**1.44 (±0.15)**
pretrain_model	21	**1.38**	**1.44 (±0.15)**
model_train	19	1.40	1.45 (±0.15)

Table 7.3: Hyperparameter search results for transfer learning method

7.2.3 Pretrained Model & Related data

As described in the previous chapter, RQ2 will be studied by a comparison of the GNN trained solely on the AbAg-Affinity dataset with a GNN utilizing a pretrained model and trained on additional related data, selected based on the results shown above. In order to find a good performing set of hyperparameters, a search was performed using the selected datasets and the pretrained model from above. Again, the best-found configuration was used in the final 10-fold cross-validation with the same splits as for RQ1.

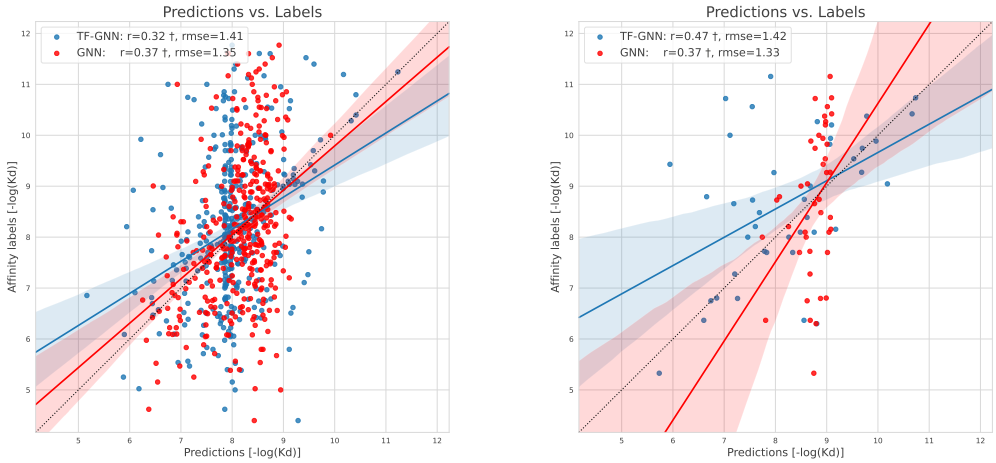
The predictions for models trained with and without a transfer learning approach are very similar (Table 7.4 & Figure 7.4). Regarding RMSE and Pearson's correlation the model trained solely on antibody-antigen data performs slightly better for the full AbAg-Affinity dataset. However, looking at the absolute errors of both approaches using the Wilcoxon signed-rank sum test, we cannot reject the possibility that the distribution of both absolute error terms has the same median (compare Figure C.4). The best results for the AB-benchmark are also not conclusive as the GNN without transfer learning has a better RMSE but a lower Pearson's correlation.

	AbAg-Affinity		AB-Benchmark	
	RMSE	Pearson's R	RMSE	Pearson's R
REF15	3.12	0.11	1.35	0.34 *
GNN	**1.82 †**	**0.37 †**	**1.33**	0.37 *
GNN + TF	1.98 †	0.32 †	1.42	**0.47 ***

Table 7.4: RMSE and Pearson's R for full AbAg-Affinity and AB-benchmark datasets.[1]
†: significant difference to REF15
*: Significance testing is not possible with a single value

Regarding robustness and stability, the GNN trained with transfer learning showed similar results as the GNN trained solely on the AbAg-Affinity dataset (Figure C.5 and Figure C.2 respectively). However, the performance of the models trained during cross-validation showed some differences for the AB-Benchmark dataset. Here the GNN without transfer learning shows more stable results for the AB-benchmark using different training splits (Figure C.3) compared to the GNN using transfer learning (Figure C.6).

[1]Table 7.4 shows the metrics on the full AbAg-Affinity dataset while Table 7.1 shows the metric average for the 10 validation sets of cross-validation



(a) AbAg-Affinity results of GNN and GNN + Transfer Learning

(b) AB-benchmark results of GNN and GNN + Transfer Learning

Figure 7.4: Predictions vs. Labels for AB-benchmark and AbAg-Affinity dataset. †: significant correlation

7.2.4 Discussion

Although integrating a pretrained model or related datasets alone seems to lead to slight improvements, the combination of pretrained models and multiple related datasets does not cause a significant improvement for predictions on the AbAg-Affinity dataset. The results of the previous section moreover indicate a decreased performance due to a loss in robustness (Figure C.6). This could be caused by a variety of reasons, like characteristical differences across models/datasets or that the model is simply not powerful enough to model similarities in these datasets (underfitting model). Future research may investigate in more detail the influence of individual datasets and pretrained models. This has the potential to lead to better knowledge transfer between related data domains and therefore increased predictive power for the antibody-antigen use case.

CHAPTER 8

Conclusion & Outlook

In this thesis, we propose an end-to-end deep learning-based approach to predict antibody-antigen binding affinity and compare it to a commonly employed baseline approach. The following paragraphs summarize the key findings by revisiting the research questions posed in the beginning, stating some limitations of the results and highlighting possible future work in this direction.

8.1 Summary & Key Findings

The search for suitable data and related methods resulted in insights into the limited availability of antibody-antigen data and a clear focus on protein-ligand binding in this domain. An analysis of the AbAg-Affinity dataset leads to similar insights as reported in [RBG+22], giving credit to the representative nature of the dataset (eg. overrepresentation of specific amino-acids in the paratope and epitope Figure A.1)

The results of this thesis allow answering the first research questions posed in Section 1.2 with high certainty.

RQ1: Does a geometric deep learning approach outperform the Rosetta Energy Function regarding the predictive power of antibody-antigen binding affinity?

The comparison included two different metrics that all showed a superiority of the GNN over the REF15 (compare Table 7.1). Using a cross-validation approach on the larger and more diverse AbAg-Affinity dataset led to significant performance differences between both approaches. On the basis of these insights, the superiority of GNNs to REF15 is concluded.

However, using a smaller subset (AB-Benchmark), selected based on restrictive criteria (see Section 4.1), no significant performance difference in absolute errors could be detected.

RQ2: Do transfer-learning strategies (domain and/or task, parallel and/or sequential) to overcome data scarcity limitations improve the predictive power of graph neural networks for antibody-antigen binding affinity prediction?

The initial evaluation of using pretrained models or related datasets did not lead to significant results and only showed that some datasets seem to slightly improve the predictive power of the GNN. A final evaluation using a promising subset of the related datasets and BindingDDG as a pretrained feature embedding model did not lead to improvements of the predictive power of the GNN.

There are several potential explanations for this result, which may include unsuitable information of the introduced datasets or an unfitting transfer-learning approach (pretraining-finetuning, bucket-training). Furthermore, the implemented GNN could lack the complexity to actually model the similarities and differences of the transfer-learning datasets or the existing noise in the validation data simply does not allow for an identification of minor improvements. Possible extensions and future research directions in this domain follow in Section 8.2.

8.1.1 Contribution

The three main contributions to the research domain of machine learning-based antibody-antigen binding affinity prediction of this thesis are summarized below.

Comprehensive preprocessing pipeline for binding affinity datasets

The datasets used throughout this thesis can be generated using Snakemake pipelines made available in addition to the source code. These workflows can also be used to generate new versions of the datasets including the latest complexes published in the respective databases.

PyTorch-based Framework for binding affinity prediction using GNNs

The source code defining the affinity prediction model used to train the model and perform the experiments is available under `https://github.com/FabianTraxler/ag_binding_affinity`. This module can be used for all binding affinity prediction tasks and is not limited to antibody-antigen data. Furthermore, the prediction pipeline (see Figure 5.3) follows a modular design and can be configured and extended in many ways.

Results for GNN on antibody-antigen data

To the best of our knowledge, this thesis is the first to evaluate the performance of a graph neural network for the antibody-antigen binding affinity prediction task, showing promising results for this application.

8.2 Limitations & Future Work

One of the main problems with deep learning-based binding affinity prediction is the scarcity of data and the quality of the available data. The quality of the 3D structure and the affinity measurements vary a lot because the currently available methods do not offer fully precise measurements [JAVH20]. This leads to noisy data that make a deep-learning approach with limited data even harder. Furthermore, this also affects the metrics and their interpretation, since we cannot assume that the affinity measurements are precise and therefore cannot expect a high correlation and low errors.

Furthermore, the selected cross-validation approach comes with the limitation that the hyperparameters were selected based on one of the validation sets (while being trained on the rest of the data). Therefore, full independence of the validation sets during cross-validation cannot be assumed.

Future Work

During the development and implementation of the GNN, many additional extensions and possible improvements emerged. These concern either the data preparation process, the GNN architecture, or the integration of related data.

Data preparation:

In this thesis, only relaxation using the Rosetta Suite was implemented, but there are many different protocols and methods available (eg. Amber Force Field [SFCW13]) that could be compared regarding their impact on GNN performance. Furthermore, by utilizing biophysical knowledge, even more features for amino acids, atoms or their connections could be provided.

GNN architecture:

While my research focussed on two well-established GNN layer architectures and aggregation methods, each step of the affinity prediction pipeline could be optimized and adapted based on the latest research on graph neural networks. For example, different message-passing layers or aggregation methods could be evaluated.

Both types of graphs (atoms and residues) could be combined, possibly leading to a better representation of the binding site using a hierarchical approach (eg. similar to An et al. [AAO+21]).

Transfer learning:

While the experiments of this thesis did not show a clear improvement using transfer-learning approaches, it is still possible that such methods are successful. For example, using a self-supervised learning task to obtain meaningful embeddings of the binding site or integrating data of structure prediction tasks are promising directions to obtain more meaningful node embeddings. Additionally, information from force fields (like energy terms) could be leveraged to increase the available information in the graph.

The implemented GNN optimized for antibody-antigen binding affinity prediction could also be used as a tool for other tasks. In contrast to previous energy-based approaches, the GNN allows for an interpretation of input effects on output values due to its "end-to-end" gradient-based implementation. Therefore, the neural network can for instance be integrated with an antibody generation process based on a specific antigen providing information on the binding strength. Additionally, in a comparison of multiple antibodies and their possible binding strength to an antigen, the GNN could also supply meaningful guidance to select the antibody with the best binding strength.

APPENDIX A

Data Analysis

Tables and plots containing information on the data used throughout this thesis.

Residue graph	
Information	Size
Amino-acid type	20
Protein type	2
Relative chain position	1
Aliphatic residue	1
Aromatic residue	1
Polar neutral residue	1
Acidic charged residue	1
basic charges residue	1
Residue weight	1
$-log(Kd)$ of $-COOH$ group	1
$-log(Kd)$ of $-NH_3$ group	1
$-log(Kd)$ of any other group	1
pH at isoelectric point	1
Hydrophobicity	2

Table A.1: Overview table of the node features for residue graphs

Atom graph	
Information	Size
Residue encoding	35
Atom type	21
Atom degree (# covalent bonds)	4
# bound H atoms	4
Implicit Valence	4
Aromatic atom	1

Table A.2: Overview table of the node features for atom graphs

Publication	**Size**	**Citation**
phillips21_bindin	26,515	Phillips et al.[PLM+21]
wu17_in	83,284	Wu et al. [WGT+17]
starr21_prosp_covid	3	Starr et al. [SGA+21]
wu20_differ_ha_h3_h1	5,765	Wu et al. [WTL+20]
mason21_optim_therap_antib_by_predic	12,528	Mason et al. [MFW+21]
taft22_deep_mutat_learn_predic_ace2	1,618,683	Taft et al. [TWG+22]
b.20_funct_screen_strat_engin_chimer	191	Di Roberto et al. [DRCRF+20]
madan21_mutat_hiv	1,035	Madan et al. [MZX+21]

Table A.3: Overview table of the publications used for the DMS dataset

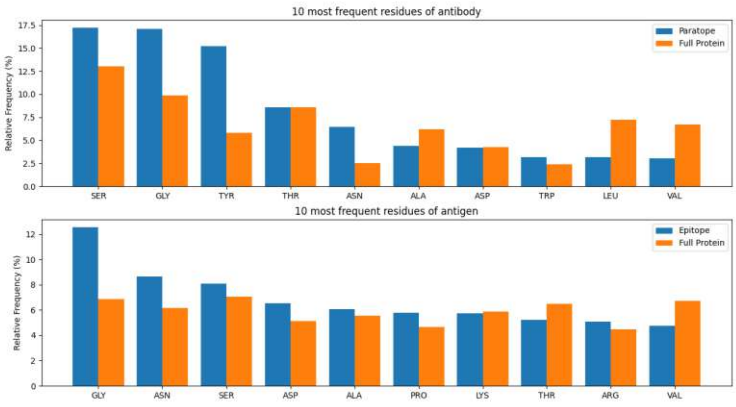


Figure A.1: Analysis of the difference of residue and atom distribution between the full protein and the binding site (top: antibody vs. paratope, bottom: antigen vs. epitope)

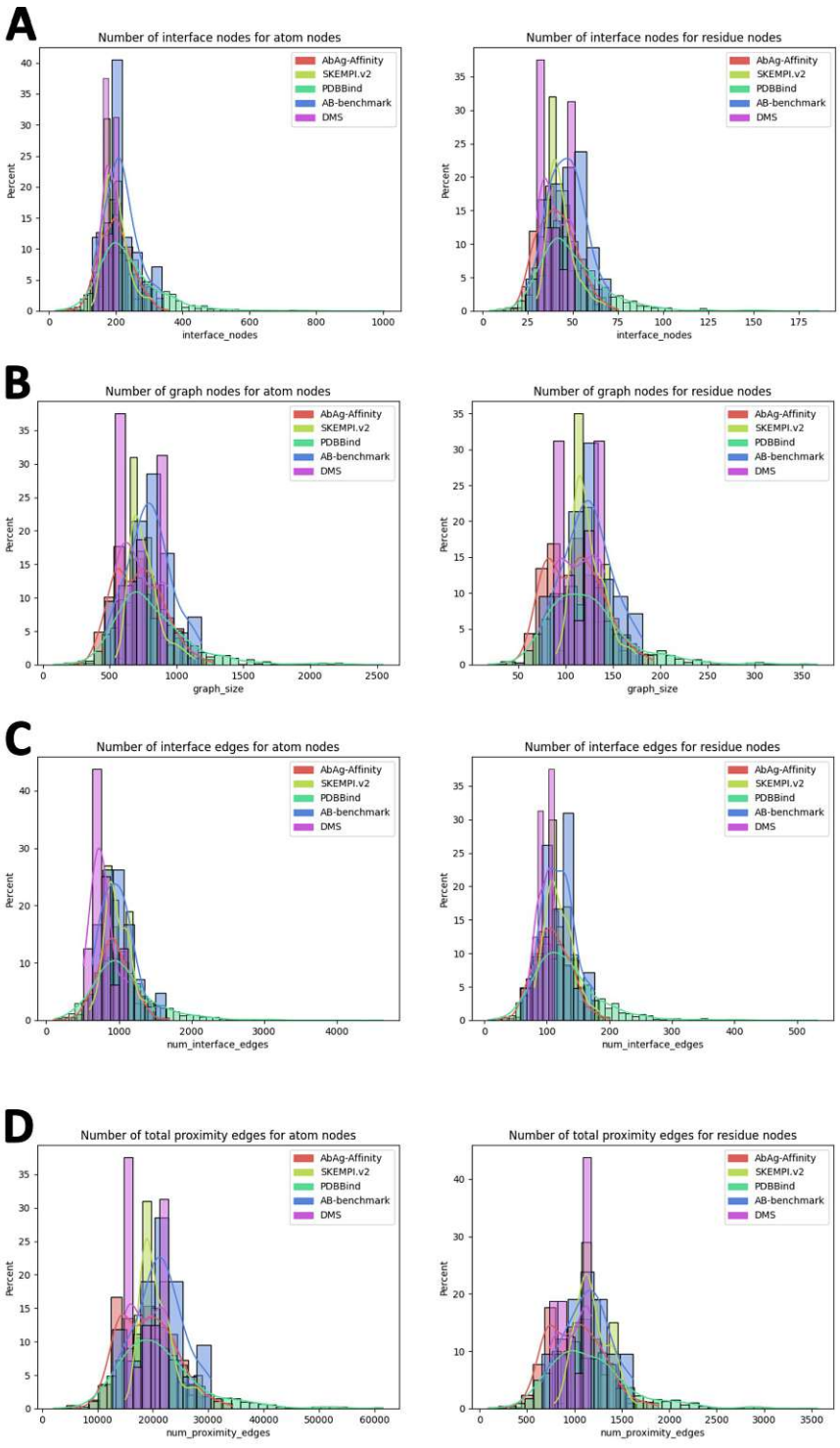


Figure A.2: Dataset comparison of graph characteristics distribution (left: atom graphs, right: residue graphs). A) Interface node count. B) Graph node count. C) Interface edge count. D) Graph edge count.

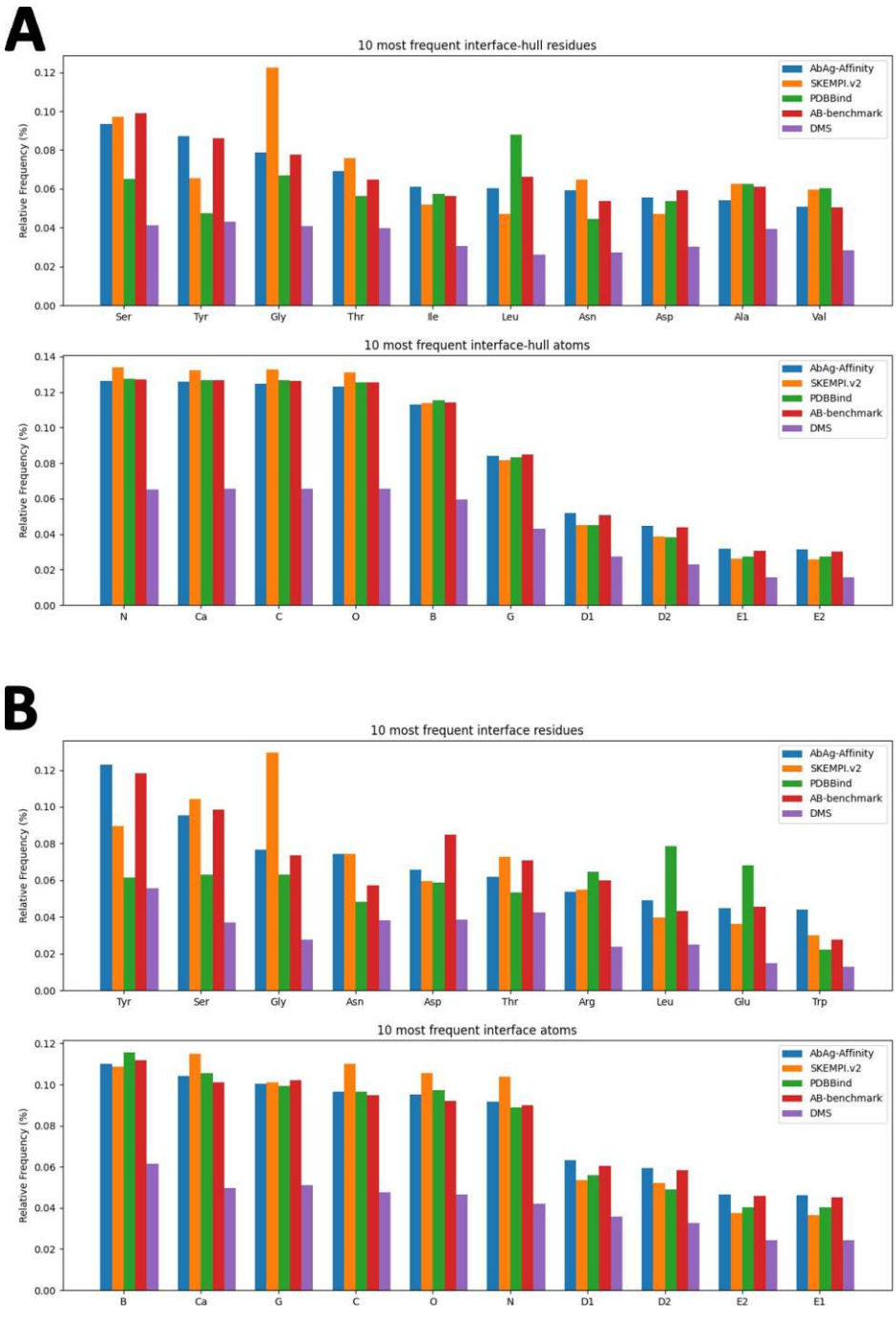


Figure A.3: Dataset comparison of node type distribution. A) Interface hull node types (top: residue graph, bottom: atom graph). B) Interface node types. (top: residue graph, bottom: atom graph)

APPENDIX B

Model Analysis

Tables and plots containing information on the GNN used throughout this thesis.

Hyperparameter	Values	Kruskal H-score	Kruskal p_value	Significant
aggregation_method	[attention, edge, max, mean]	110.4186	0.0000	True
attention_heads	[1, 3, 5]	5.7417	0.0566	False
batch_size	[1, 5, 10]	0.7823	0.6763	False
channel_halving	[False, True]	1.2195	0.2695	False
max_num_nodes	[10, 50, None]	0.3508	0.8391	False
node_type	[atom, residue]	0.0052	0.9424	False
loss_function	[L1, L2]	4.6160	0.0317	True
layer_type	[GAT, GCN]	0.0160	0.8993	False
gnn_type	[guided, proximity]	2.5989	0.1069	False
num_gnn_layers	[0, 3, 5]	4.4728	0.1068	False
num_fc_layers	[1, 5, 10]	9.2288	0.0099	True
fc_size_halving	[False, True]	2.4501	0.1175	False
nonlinearity	[gelu, leaky, relu]	2.8163	0.2446	False
relaxed_pdbs	[False, True]	2.7303	0.0985	False

Table B.1: Overview table of the GNN hyperparameter and their impact on model performance

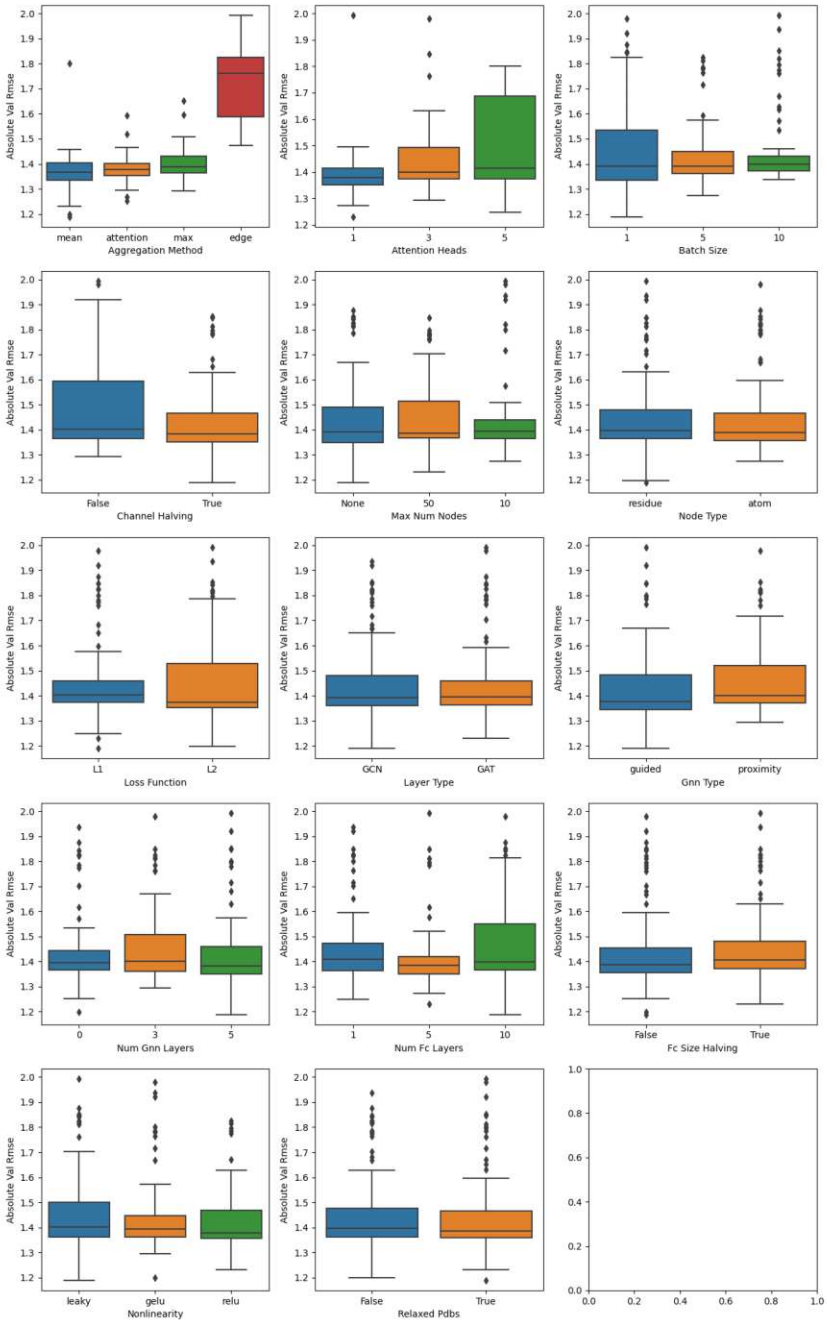


Figure B.1: Overview of the results for each hyperparameter value (Only runs with $RSME \leq 2$ shown)

APPENDIX C

Experiments

Parameter	Value
node_type	residue
batch_size	1
layer_type	GCN
nonlinearity	leaky
relaxed_pdbs	true
scale_values	true
learning_rate	0.0002374
loss_function	L1
max_num_nodes	None
num_fc_layers	10
num_gnn_layers	5
channel_halving	true
fc_size_halving	false
max_edge_distance	3
aggregation_method	mean

Table C.1: Table showing the best-performing hyperparameter configuration of the hyperparameter search

Hyperparameter	Values	Kruskal H-score	Kruskal p_value	Significant
pretrained _model	Binding_DDG, DeepRefine, No-Model	1.993466	0.369083	False
transfer _learning _dataset	DMS-madan21_mutat _hiv:relative, DMS-mason21_comb_optim _therap_antib_by_predic _combined_H3_3:relative, DMS-mason21_comb_optim _therap_antib_by_predic _combined_L3_3:relative, DMS-mason21_optim_therap _antib_by_predic _dms_H:relative, DMS-mason21_optim_therap _antib_by_predic _dms_L:relative, DMS-phillips21_bindin:absolute, DMS-phillips21_bindin:relative, DMS-wu17_in:relative, PDBBind:absolute, SKEMPI.v2:absolute, SKEMPI.v2:relative, no_dataset	10.69	0.56	False
train _strategy	bucket_train, model_train, pretrain_model	1.54	0.46	False

Table C.2: Overview table of the pretrained model and related data comparison and their impact on model performance

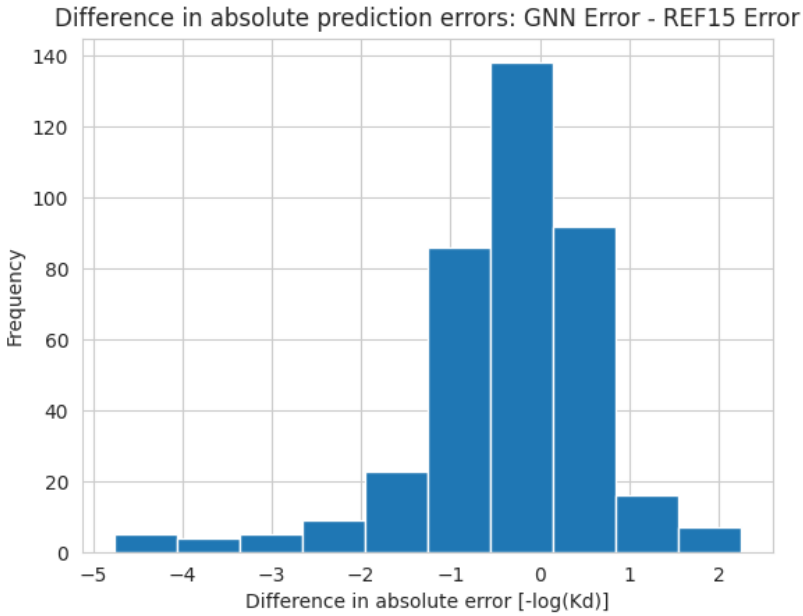


Figure C.1: Histogram of differences (GNN error - REF15 error) in absolute errors

transfer_learning_datasets	runs	median	mean	std
DMS-mason21_optim_therap_antib _by_predic_dms_L:relative	3.0	1.336388	1.384754	0.083933
DMS-phillips21_bindin:relative	1.0	1.352681	1.352681	NaN
DMS-mason21_optim_therap_antib _by_predic_dms_H:relative	3.0	1.365688	1.365711	0.018163
PDBBind:absolute	3.0	1.372487	1.361140	0.020919
SKEMPI.v2:relative	3.0	1.375937	1.546211	0.326679
SKEMPI.v2:absolute	7.0	1.378059	1.455285	0.179579
DMS-b.20_funct_screen_strat_engin _chimer:relative	3.0	1.390536	1.405485	0.124582
DMS-madan21_mutat_hiv:relative	6.0	1.394934	1.417177	0.114558
no_dataset	19.0	1.400437	1.454567	0.148682
DMS-mason21_comb_optim_therap_antib _by_predic_combined_L3:relativ	3.0	1.404235	1.403509	0.027441
DMS-mason21_comb_optim_therap_antib _by_predic_combined_H3:relativ	4.0	1.457334	1.503754	0.173792
DMS-phillips21_bindin:absolute	2.0	1.479902	1.479902	0.138483
DMS-wu17_in:relative	2.0	1.597617	1.597617	0.176789

Table C.3: Complete hyperparameter search results for related data

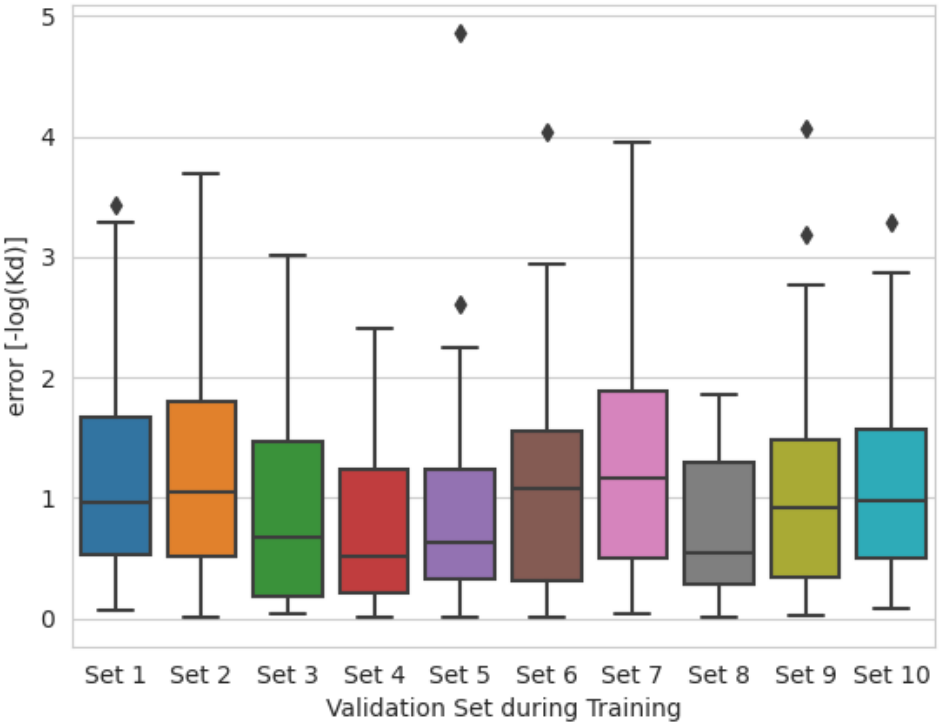


Figure C.2: BoxPlot of absolute errors on the 10 different AbAg-Affinity validation sets during the cross-validation

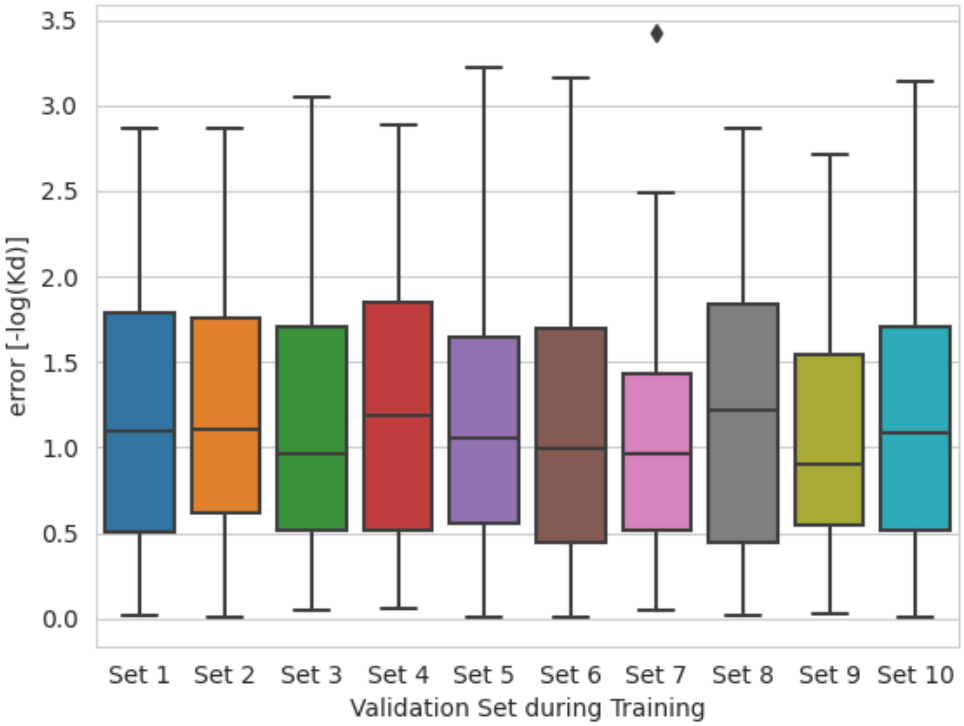


Figure C.3: BoxPlot of absolute errors on the AB-benchmark for 10 different training dataset combinations during cross-validation

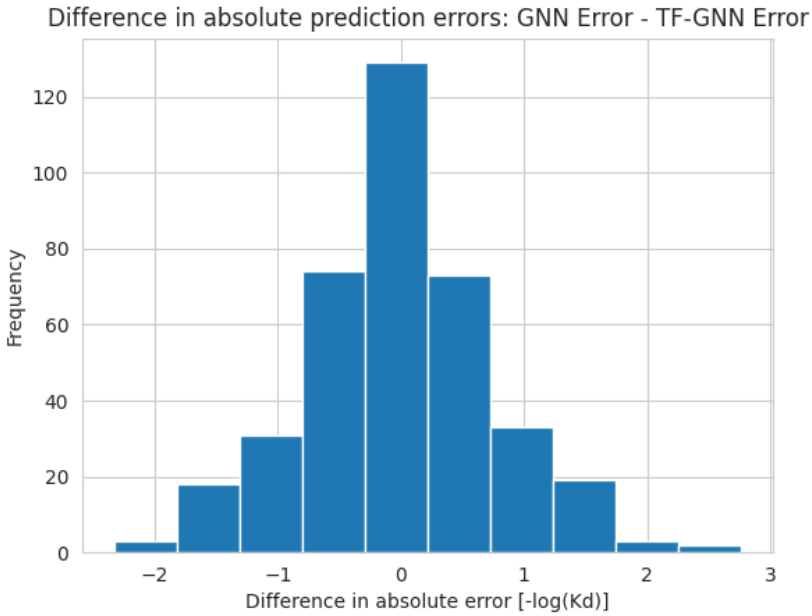


Figure C.4: Histogram of differences (GNN error - GNN with transfer learning error) in absolute errors

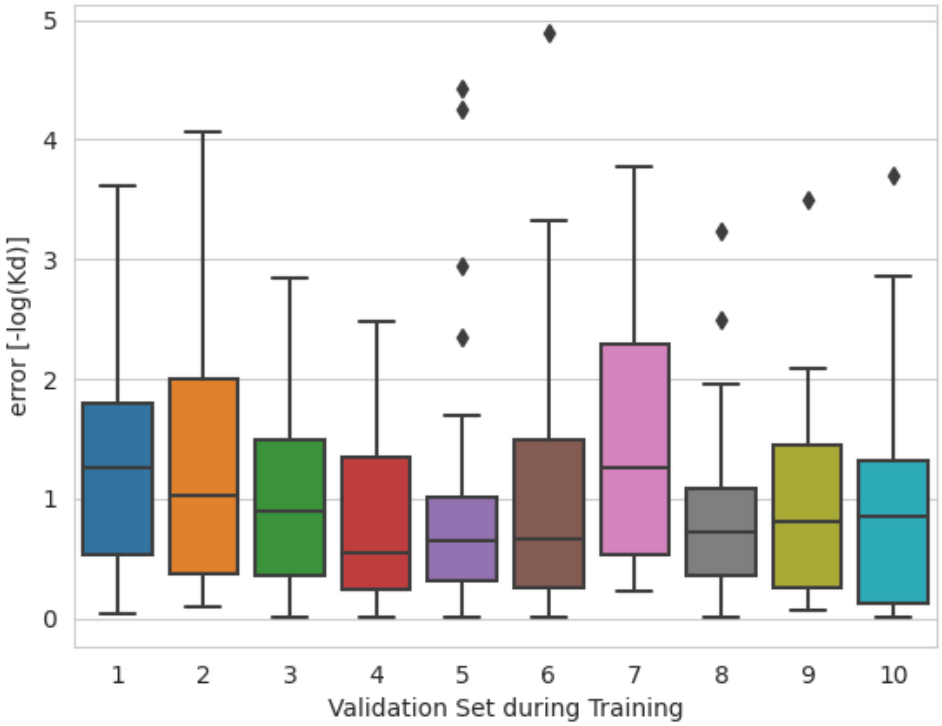


Figure C.5: BoxPlot of absolute errors on the 10 different AbAg-Affinity validation sets during the cross-validation using the transfer-learning appracoh

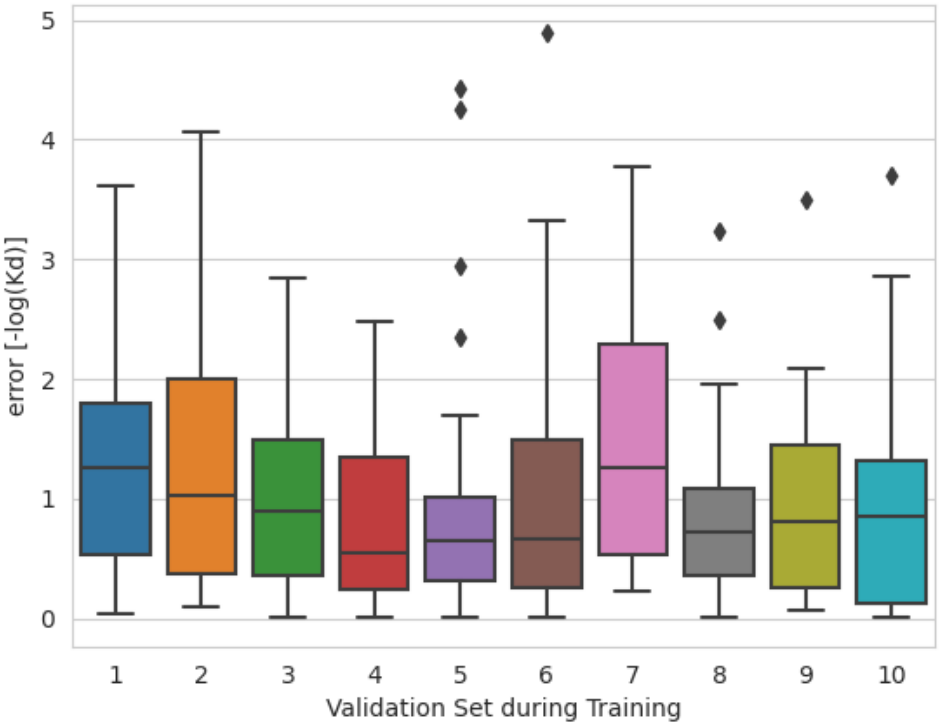


Figure C.6: BoxPlot of absolute errors on the AB-benchmark for 10 different training dataset combinations during cross-validation using the transfer-learning approach

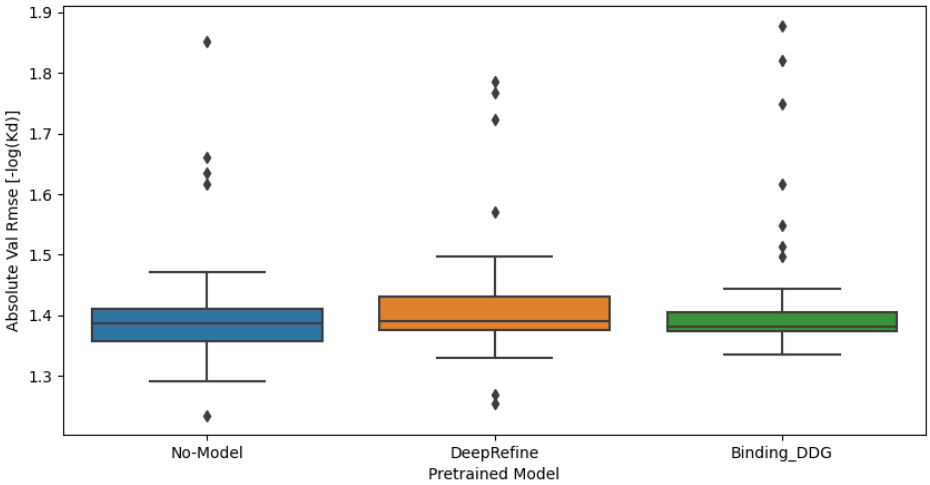


Figure C.7: Performance of the pretrained models compared with no pretrained model using random hyperparameter configurations

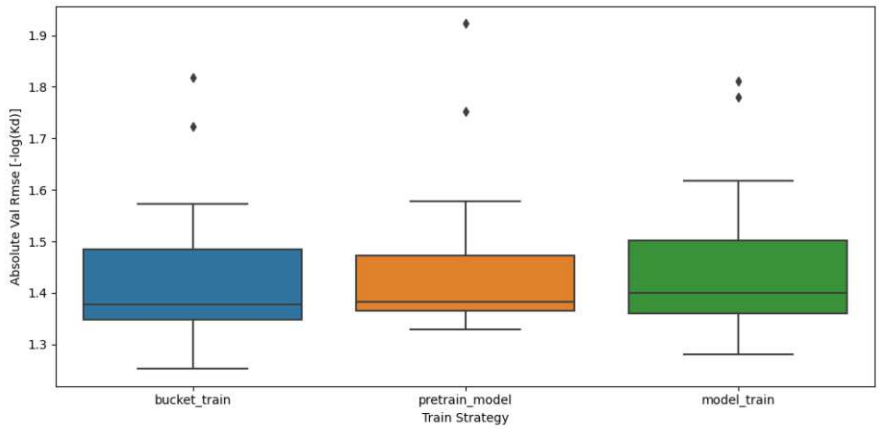


Figure C.8: Performance of different training strategies using random hyperparameter configurations

List of Figures

List of Tables

Bibliography

[AAO+21] Guangzhou An, Masahiro Akiba, Kazuko Omodaka, Toru Nakazawa, and Hideo Yokota. Hierarchical deep learning models using transfer learning for disease detection and classification based on small number of medical images. *Scientific Reports*, 11(1):4250, March 2021. Number: 1 Publisher: Nature Publishing Group.

[AF11] Carlos L. Araya and Douglas M. Fowler. Deep mutational scanning: assessing protein function on a massive scale. *Trends in Biotechnology*, 29(9):435–442, September 2011.

[AGM+90] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, October 1990.

[ALFJ+17] Rebecca F. Alford, Andrew Leaver-Fay, Jeliazko R. Jeliazkov, Matthew J. O'Meara, Frank P. DiMaio, Hahnbeom Park, Maxim V. Shapovalov, P. Douglas Renfrew, Vikram K. Mulligan, Kalli Kappel, Jason W. Labonte, Michael S. Pacella, Richard Bonneau, Philip Bradley, Roland L. Jr. Dunbrack, Rhiju Das, David Baker, Brian Kuhlman, Tanja Kortemme, and Jeffrey J. Gray. The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design. *Journal of Chemical Theory and Computation*, 13(6):3031–3048, June 2017. Publisher: American Chemical Society.

[BAY22] Shaked Brody, Uri Alon, and Eran Yahav. How Attentive are Graph Attention Networks?, January 2022. arXiv:2105.14491 [cs].

[BBCV21] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, May 2021. arXiv:2104.13478 [cs, stat].

[BBL+17] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, July 2017. Conference Name: IEEE Signal Processing Magazine.

[BGM+88] G. Boulot, V. Guillon, R. A. Mariuzza, R. J. Poljak, M. M. Riottot, H. Souchon, S. Spinelli, and D. Tello. Crystallization of antibody fragments and their complexes with antigen. *Journal of Crystal Growth*, 90(1):213–221, July 1988.

[Bie20] Lukas Biewald. Experiment Tracking with Weights and Biases, 2020.

[BLW76] Norman L. Biggs, E. Keith Lloyd, and Robin J. Wilson. *Graph theory 1736–1936*. Clarendon Press; Oxford University Press, London, 1976. Published: London: Clarendon Press; Oxford University Press. X, 239 p. (1976).

[BWF+00] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, January 2000.

[CA06] Max D. Cooper and Matthew N. Alder. The Evolution of Adaptive Immune Systems. *Cell*, 124(4):815–822, February 2006.

[CAC+09] Peter JA Cock, Tiago Antao, Jeffrey T Chang, Brad A Chapman, Cymon J Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and others. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, 2009. Publisher: Oxford University Press.

[Car93] Richard A. Caruana. Multitask Learning: A Knowledge-Based Source of Inductive Bias. In *Machine Learning Proceedings 1993*, pages 41–48. Morgan Kaufmann, San Francisco (CA), January 1993.

[CTD+14] Patrick Conway, Michael D. Tyka, Frank DiMaio, David E. Konerding, and David Baker. Relaxation of backbone bond geometry improves protein energy landscape modeling. *Protein Science*, 23(1):47–55, 2014. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/pro.2389.

[Dem06] Janez Demšar. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006.

[DGKP14] Michael Drmota, Bernhard Gittenberger, Günther Karigl, and Alois Panholzer. *Mathematik für Informatik. Vierte erweiterte Auflage*. Heldermann Verlag, 2014. Accepted: 2022-07-29T08:13:19Z.

[DKL+14] James Dunbar, Konrad Krawczyk, Jinwoo Leem, Terry Baker, Angelika Fuchs, Guy Georges, Jiye Shi, and Charlotte M. Deane. SAbDab: the structural antibody database. *Nucleic Acids Research*, 42(Database issue):D1140–1146, January 2014.

[DLX+16] Xing Du, Yi Li, Yuan-Ling Xia, Shi-Meng Ai, Jing Liang, Peng Sang, Xing-Lai Ji, and Shu-Qun Liu. Insights into Protein–Ligand Interactions: Mechanisms, Models, and Methods. *International Journal of Molecular Sciences*, 17(2):144, February 2016. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[DRCRF+20] Raphaël B. Di Roberto, Rocío Castellanos-Rueda, Samara Frey, David Egli, Rodrigo Vazquez-Lombardi, Edo Kapetanovic, Jakub Kucharczyk, and Sai T. Reddy. A Functional Screening Strategy for Engineering Chimeric Antigen Receptors with Reduced On-Target, Off-Tumor Activation. *Molecular Therapy*, 28(12):2564–2576, December 2020.

[Dun64] Olive Jean Dunn. Multiple Comparisons Using Rank Sums. *Technometrics*, 6(3):241–252, August 1964. Publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/00401706.1964.10490181.

[FM18] Saba Ferdous and Andrew C R Martin. AbDb: antibody structure database—a database of PDB-derived antibody structures. *Database*, 2018:bay040, January 2018.

[FPP07] David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

[FPRA20] Abolfazl Farahani, Behrouz Pourshojae, Khaled Rasheed, and Hamid R. Arabnia. A Concise Review of Transfer Learning. In *2020 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 344–351, December 2020.

[Fre] Mathias Frey. PyTorch Geometric.

[FSW+18] Evan N. Feinberg, Debnil Sur, Zhenqin Wu, Brooke E. Husic, Huanghao Mai, Yang Li, Saisai Sun, Jianyi Yang, Bharath Ramsundar, and Vijay S. Pande. PotentialNet for Molecular Property Prediction. *ACS Central Science*, 4(11):1520–1530, November 2018. Publisher: American Chemical Society.

[GEW06] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, April 2006.

[GPC+05] Sungsam Gong, Changbum Park, Hansol Choi, Junsu Ko, Insoo Jang, Jungsul Lee, Dan M. Bolser, Donghoon Oh, Deok-Soo Kim, and Jong Bhak. A protein domain interaction interface database: InterPare. *BMC Bioinformatics*, 6(1):207, August 2005.

[Gre] Rachel Green. PDB101: Learn: Guide to Understanding PDB Data: Beginner's Guide to PDB Structures and the PDBx/mmCIF Format.

[GVZ+21] Johnathan D. Guest, Thom Vreven, Jing Zhou, Iain Moal, Jeliazko R. Jeliazkov, Jeffrey J. Gray, Zhiping Weng, and Brian G. Pierce. An expanded benchmark for antibody-antigen docking and affinity prediction reveals insights into antibody recognition determinants. *Structure (London, England: 1993)*, 29(6):606–621.e5, June 2021.

[HCH+21] Yunfei Hu, Kai Cheng, Lichun He, Xu Zhang, Bin Jiang, Ling Jiang, Conggang Li, Guan Wang, Yunhuang Yang, and Maili Liu. NMR-Based Methods for Protein Analysis. *Analytical Chemistry*, 93(4):1866–1879, February 2021. Publisher: American Chemical Society.

[HMW+22] Kyrin R. Hanning, Mason Minot, Annmaree K. Warrender, William Kelton, and Sai T. Reddy. Deep mutational scanning for therapeutic antibody engineering. *Trends in Pharmacological Sciences*, 43(2):123–135, February 2022.

[Hod22] Timothy O. Hodson. Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not. *Geoscientific Model Development*, 15(14):5481–5487, July 2022. Publisher: Copernicus GmbH.

[Jan01] Charles Janeway. *Immunobiology 5: the immune system in health and disease*. Garland Pub., New York, 2001. OCLC: 45708106.

[JAVH20] Inga Jarmoskaite, Ishraq AlSadhan, Pavanapuresan P Vaidyanathan, and Daniel Herschlag. How to measure and evaluate binding affinities. *eLife*, 9:e57264, August 2020. Publisher: eLife Sciences Publications, Ltd.

[JEP+21] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021. Number: 7873 Publisher: Nature Publishing Group.

[JJGD+19] Justina Jankauskaitė, Brian Jiménez-García, Justas Dapkūnas, Juan Fernández-Recio, and Iain H Moal. SKEMPI 2.0: an updated benchmark of changes in protein–protein binding energy, kinetics and thermodynamics upon mutation. *Bioinformatics*, 35(3):462–469, February 2019.

[JKZ+21] Derek Jones, Hyojin Kim, Xiaohua Zhang, Adam Zemla, Garrett Stevenson, W. F. Drew Bennett, Daniel Kirshner, Sergio E. Wong, Felice C.

Lightstone, and Jonathan E. Allen. Improved Protein–Ligand Binding Affinity Prediction with Structure-Based Deep Fusion Inference. *Journal of Chemical Information and Modeling*, 61(4):1583–1592, April 2021. Publisher: American Chemical Society.

[JLGWS21] José Jiménez-Luna, Francesca Grisoni, Nils Weskamp, and Gisbert Schneider. Artificial intelligence in drug discovery: recent advances and future perspectives. *Expert Opinion on Drug Discovery*, 16(9):949–959, September 2021. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/17460441.2021.1909567.

[JLZ+20] Mingjian Jiang, Zhen Li, Shugang Zhang, Shuang Wang, Xiaofeng Wang, Qing Yuan, and Zhiqiang Wei. Drug–target affinity prediction using graph neural network and contact maps. *RSC Advances*, 10(35):20701–20712, May 2020. Publisher: The Royal Society of Chemistry.

[KRKP+16] Thomas Kluyver, Benjamin Ragan-Kelley, P&#233, Fernando Rez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Dami&#225 Avila, n, Safia Abdalla, Carol Willing, and Jupyter Development Team. Jupyter Notebooks – a publishing format for reproducible computational workflows. *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87–90, 2016. Publisher: IOS Press.

[KW52] William H. Kruskal and W. Allen Wallis. Use of Ranks in One-Criterion Variance Analysis. *Journal of the American Statistical Association*, 47(260):583–621, December 1952. Publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1952.10483441.

[KW17] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, February 2017. arXiv:1609.02907 [cs, stat].

[Lea01] Andrew R. Leach. *Molecular modelling : principles and applications*. Pearson, Harlow, England ; New York : Prentice Hall, 2 edition, 2001.

[LGM15] Pedro E. M. Lopes, Olgun Guvench, and Alexander D. MacKerell. Current Status of Protein Force Fields for Molecular Dynamics Simulations. In Andreas Kukol, editor, *Molecular Modeling of Proteins*, Methods in Molecular Biology, pages 47–71. Springer, New York, NY, 2015.

[LSJ08] Nan Li, Zhonghua Sun, and Fan Jiang. Prediction of protein-protein binding site by using core interface residue and support vector machine. *BMC Bioinformatics*, 9(1):553, December 2008.

[LWL+20] Julia Koehler Leman, Brian D. Weitzner, Steven M. Lewis, Jared Adolf-Bryfogle, Nawsad Alam, Rebecca F. Alford, Melanie Aprahamian, David Baker, Kyle A. Barlow, Patrick Barth, Benjamin Basanta, Brian J. Bender,

Kristin Blacklock, Jaume Bonet, Scott E. Boyken, Phil Bradley, Chris Bystroff, Patrick Conway, Seth Cooper, Bruno E. Correia, Brian Coventry, Rhiju Das, René M. De Jong, Frank DiMaio, Lorna Dsilva, Roland Dunbrack, Alexander S. Ford, Brandon Frenz, Darwin Y. Fu, Caleb Geniesse, Lukasz Goldschmidt, Ragul Gowthaman, Jeffrey J. Gray, Dominik Gront, Sharon Guffy, Scott Horowitz, Po-Ssu Huang, Thomas Huber, Tim M. Jacobs, Jeliazko R. Jeliazkov, David K. Johnson, Kalli Kappel, John Karanicolas, Hamed Khakzad, Karen R. Khar, Sagar D. Khare, Firas Khatib, Alisa Khramushin, Indigo C. King, Robert Kleffner, Brian Koepnick, Tanja Kortemme, Georg Kuenze, Brian Kuhlman, Daisuke Kuroda, Jason W. Labonte, Jason K. Lai, Gideon Lapidoth, Andrew Leaver-Fay, Steffen Lindert, Thomas Linsky, Nir London, Joseph H. Lubin, Sergey Lyskov, Jack Maguire, Lars Malmström, Enrique Marcos, Orly Marcu, Nicholas A. Marze, Jens Meiler, Rocco Moretti, Vikram Khipple Mulligan, Santrupti Nerli, Christoffer Norn, Shane Ó'Conchúir, Noah Ollikainen, Sergey Ovchinnikov, Michael S. Pacella, Xingjie Pan, Hahnbeom Park, Ryan E. Pavlovicz, Manasi Pethe, Brian G. Pierce, Kala Bharath Pilla, Barak Raveh, P. Douglas Renfrew, Shourya S. Roy Burman, Aliza Rubenstein, Marion F. Sauer, Andreas Scheck, William Schief, Ora Schueler-Furman, Yuval Sedan, Alexander M. Sevy, Nikolaos G. Sgourakis, Lei Shi, Justin B. Siegel, Daniel-Adriano Silva, Shannon Smith, Yifan Song, Amelie Stein, Maria Szegedy, Frank D. Teets, Summer B. Thyme, Ray Yu-Ruei Wang, Andrew Watkins, Lior Zimmerman, and Richard Bonneau. Macromolecular modeling and design in Rosetta: recent methods and frameworks. *Nature Methods*, 17(7):665–680, July 2020. Number: 7 Publisher: Nature Publishing Group.

[MCW+22] Alex Morehead, Xiao Chen, Tianqi Wu, Jian Liu, and Jianlin Cheng. EGR: Equivariant Graph Refinement and Assessment of 3D Protein Complex Structures, May 2022. arXiv:2205.10390 [cs, q-bio, stat].

[MFW+21] Derek M. Mason, Simon Friedensohn, Cédric R. Weber, Christian Jordi, Bastian Wagner, Simon M. Meng, Roy A. Ehling, Lucia Bonati, Jan Dahinden, Pablo Gainza, Bruno E. Correia, and Sai T. Reddy. Optimization of therapeutic antibodies by predicting antigen specificity from antibody sequence via deep learning. *Nature Biomedical Engineering*, 5(6):600–612, June 2021. Number: 6 Publisher: Nature Publishing Group.

[MHN13] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning*, volume 28, page 3. Atlanta, Georgia, USA, 2013. Issue: 1.

[MJL+21] Felix Mölder, Kim Philipp Jablonski, Brice Letcher, Michael B. Hall, Christopher H. Tomkins-Tinch, Vanessa Sochat, Jan Forster, Soohyun Lee,

Sven O. Twardziok, Alexander Kanitz, Andreas Wilm, Manuel Holtgrewe, Sven Rahmann, Sven Nahnsen, and Johannes Köster. Sustainable data analysis with Snakemake. Technical Report 10:33, F1000Research, January 2021. Type: article.

[MPA22] Yoochan Myung, Douglas E V Pires, and David B Ascher. CSM-AB: graph-based antibody–antigen binding affinity prediction and docking scoring function. *Bioinformatics*, 38(4):1141–1143, February 2022.

[MZX+21] Bharat Madan, Baoshan Zhang, Kai Xu, Cara W. Chao, Sijy O'Dell, Jacy R. Wolfe, Gwo-Yu Chuang, Ahmed S. Fahad, Hui Geng, Rui Kong, Mark K. Louder, Thuy Duong Nguyen, Reda Rawi, Arne Schön, Zizhang Sheng, Rajani Nimrania, Yiran Wang, Tongqing Zhou, Bob C. Lin, Nicole A. Doria-Rose, Lawrence Shapiro, Peter D. Kwong, and Brandon J. DeKosky. Mutational fitness landscapes reveal genetic and structural improvement pathways for a vaccine-elicited HIV-1 broadly neutralizing antibody. *Proceedings of the National Academy of Sciences*, 118(10):e2011653118, March 2021. Publisher: Proceedings of the National Academy of Sciences.

[NAB+20] Richard A Norman, Francesco Ambrosetti, Alexandre M J J Bonvin, Lucy J Colwell, Sebastian Kelm, Sandeep Kumar, and Konrad Krawczyk. Computational approaches to therapeutic antibody design: established methods and emerging trends. *Briefings in Bioinformatics*, 21(5):1549–1567, September 2020.

[noa22] Python Language Reference, version 3.8.13, March 2022.

[NT03] Irene M.A. Nooren and Janet M. Thornton. Diversity of protein–protein interactions. *The EMBO Journal*, 22(14):3486–3492, July 2003. Publisher: John Wiley & Sons, Ltd.

[PA16] Douglas E.V. Pires and David B. Ascher. mCSM-AB: a web server for predicting antibody–antigen affinity changes upon mutation with graph-based signatures. *Nucleic Acids Research*, 44(W1):W469–W473, July 2016.

[PLJY14] Hung-Pin Peng, Kuo Hao Lee, Jhih-Wei Jian, and An-Suei Yang. Origins of specificity and affinity in antibody–protein interactions. *Proceedings of the National Academy of Sciences*, 111(26):E2656–E2665, July 2014. Publisher: Proceedings of the National Academy of Sciences.

[PLM+21] Angela M Phillips, Katherine R Lawrence, Alief Moulana, Thomas Dupic, Jeffrey Chang, Milo S Johnson, Ivana Cvijovic, Thierry Mora, Aleksandra M Walczak, and Michael M Desai. Binding affinity landscapes constrain the evolution of broadly neutralizing anti-influenza antibodies. *eLife*, 10:e71393, September 2021. Publisher: eLife Sciences Publications, Ltd.

[PLT01] Jong Park, Michael Lappe, and Sarah A Teichmann. Mapping protein family interactions: intramolecular and intermolecular protein family interaction repertoires in the PDB and yeast11Edited by J. Karn. *Journal of Molecular Biology*, 307(3):929–938, March 2001.

[Pol10] Thomas D. Pollard. A Guide to Simple and Informative Binding Assays. *Molecular Biology of the Cell*, 21(23):4061–4067, December 2010. Publisher: American Society for Cell Biology (mboc).

[Ras17] Sebastian Raschka. BioPandas: Working with molecular structures in pandas DataFrames. *Journal of Open Source Software*, 2(14):279, June 2017.

[RBG+22] Pedro B. P. S. Reis, German P. Barletta, Luca Gagliardi, Sara Fortuna, Miguel A. Soler, and Walter Rocchia. Antibody-Antigen Binding Interface Analysis in the Big Data Era. *Frontiers in Molecular Biosciences*, 9, 2022.

[RBN+18] Aliza B. Rubenstein, Kristin Blacklock, Hai Nguyen, David A. Case, and Sagar D. Khare. Systematic Comparison of Amber and Rosetta Energy Functions for Protein Structure Evaluation. *Journal of Chemical Theory and Computation*, 14(11):6015–6025, November 2018. Publisher: American Chemical Society.

[RTTB18] João P. G. L. M. Rodrigues, João M. C. Teixeira, Mikaël Trellet, and Alexandre M. J. J. Bonvin. pdb-tools: a swiss army knife for molecular structures. Technical Report 7:1961, F1000Research, December 2018. Type: article.

[Sch] Schrödinger, LLC. The PyMOL Molecular Graphics System.

[SCKO13] Inbal Sela-Culang, Vered Kunik, and Yanay Ofran. The Structural Basis of Antibody-Antigen Recognition. *Frontiers in Immunology*, 4, 2013.

[SDW+20] Chao Shen, Junjie Ding, Zhe Wang, Dongsheng Cao, Xiaoqin Ding, and Tingjun Hou. From machine learning to deep learning: Advances in scoring functions for protein–ligand docking. *WIREs Computational Molecular Science*, 10(1):e1429, 2020. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1429.

[SFCW13] Romelia Salomon-Ferrer, David A. Case, and Ross C. Walker. An overview of the Amber biomolecular simulation package. *WIREs Computational Molecular Science*, 3(2):198–210, 2013. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1121.

[SFW93] Enrico A. Stura, Gail G. Fieser, and Ian A. Wilson. Crystallization of Antibodies and Antibody-Antigen Complexes. *ImmunoMethods*, 3(3):164–179, December 1993.

[SGA+21] Tyler N. Starr, Allison J. Greaney, Amin Addetia, William W. Hannon, Manish C. Choudhary, Adam S. Dingens, Jonathan Z. Li, and Jesse D. Bloom. Prospective mapping of viral mutations that escape antibodies used to treat COVID-19. *Science*, 371(6531):850–854, February 2021. Publisher: American Association for the Advancement of Science.

[SLY+22] Sisi Shan, Shitong Luo, Ziqing Yang, Junxian Hong, Yufeng Su, Fan Ding, Lili Fu, Chenyu Li, Peng Chen, Jianzhu Ma, Xuanling Shi, Qi Zhang, Bonnie Berger, Linqi Zhang, and Jian Peng. Deep learning guided optimization of human antibody against SARS-CoV-2 variants with broad neutralization. *Proceedings of the National Academy of Sciences*, 119(11):e2122954119, March 2022. Publisher: Proceedings of the National Academy of Sciences.

[SM00] M. S. Smyth and J. H. J. Martin. x Ray crystallography. *Molecular Pathology*, 53(1):8–14, February 2000. Publisher: BMJ Publishing Group Ltd.

[SYL78] J. J. SYLVESTER. Chemistry and Algebra. *Nature*, 17(432):284–284, February 1878.

[SZ20] Till Siebenmorgen and Martin Zacharias. Computational prediction of protein–protein binding affinities. *WIREs Computational Molecular Science*, 10(3):e1448, 2020. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1448.

[TRA+19] Maha Thafar, Arwa Bin Raies, Somayah Albaradei, Magbubah Essack, and Vladimir B. Bajic. Comparison Study of Computational Prediction Tools for Drug-Target Binding Affinities. *Frontiers in Chemistry*, 7, 2019.

[TWG+22] Joseph M. Taft, Cédric R. Weber, Beichen Gao, Roy A. Ehling, Jiami Han, Lester Frei, Sean W. Metcalfe, Max D. Overath, Alexander Yermanos, William Kelton, and Sai T. Reddy. Deep mutational learning predicts ACE2 binding and antibody escape to combinatorial mutations in the SARS-CoV-2 receptor-binding domain. *Cell*, 185(21):4008–4022.e14, October 2022.

[VCC+18] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018. arXiv:1710.10903 [cs, stat].

[Wan20] Renxiao Wang. The PDBbind-CN database, 2020.

[WFLW04] Renxiao Wang, Xueliang Fang, Yipin Lu, and Shaomeng Wang. The PDBbind Database: Collection of Binding Affinities for ProteinLigand Complexes with Known Three-Dimensional Structures. *Journal of Medicinal Chemistry*, 47(12):2977–2980, June 2004. Publisher: American Chemical Society.

[WGT+17] Nicholas C. Wu, Geramie Grande, Hannah L. Turner, Andrew B. Ward, Jia Xie, Richard A. Lerner, and Ian A. Wilson. In vitro evolution of an influenza broadly neutralizing antibody is modulated by hemagglutinin receptor specificity. *Nature Communications*, 8(1):15371, May 2017. Number: 1 Publisher: Nature Publishing Group.

[Wil45] Frank Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945. Publisher: [International Biometric Society, Wiley].

[WPC+21] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, January 2021. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[WTL+20] Nicholas C. Wu, Andrew J. Thompson, Juhye M. Lee, Wen Su, Britni M. Arlian, Jia Xie, Richard A. Lerner, Hui-Ling Yen, Jesse D. Bloom, and Ian A. Wilson. Different genetic barriers for resistance to HA stem antibodies in influenza H3 and H1 viruses. *Science*, 368(6497):1335–1340, June 2020. Publisher: American Association for the Advancement of Science.

[WZL+21] Zechen Wang, Liangzhen Zheng, Yang Liu, Yuanyuan Qu, Yong-Qiang Li, Mingwen Zhao, Yuguang Mu, and Weifeng Li. OnionNet-2: A Convolutional Neural Network Model for Predicting Protein-Ligand Binding Affinity Based on Residue-Atom Contacting Shells. *Frontiers in Chemistry*, 9, 2021.

[YJG03] Andy B. Yoo, Morris A. Jette, and Mark Grondona. SLURM: Simple Linux Utility for Resource Management. In Dror Feitelson, Larry Rudolph, and Uwe Schwiegelshohn, editors, *Job Scheduling Strategies for Parallel Processing*, Lecture Notes in Computer Science, pages 44–60, Berlin, Heidelberg, 2003. Springer.

[ZCH+20] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, January 2020.

[ZZW+22] Lingling Zhao, Yan Zhu, Junjie Wang, Naifeng Wen, Chunyu Wang, and Liang Cheng. A brief review of protein–ligand interaction prediction. *Computational and Structural Biotechnology Journal*, 20:2831–2838, January 2022.