

Antibody structure prediction using deep learning



Brennan Abanades
St Cross College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2023

Abstract

The aim of this doctoral project was to improve the prediction of antibody structures and optimise the use of existing antibody data. As a starting point, we developed ABlooper [1], a tool that predicts the position of backbone atoms for CDR loops in antibodies. In Chapter 2, we present the methods and results for ABlooper. This tool has enabled the modelling of antibody structures at previously unprecedented speeds, predicting the CDR loops for one hundred structures in less than five seconds. However, ABlooper has some limitations: it cannot model side-chain atoms and sometimes generates unphysical structures, which need to be corrected using computationally expensive methods.

To overcome these limitations, we developed ImmuneBuilder [2], a novel method based on the recently released AlphaFold2 models [3]. ImmuneBuilder is a set of deep learning-based tools for modelling antibodies, nanobodies, and TCRs. In contrast to ABlooper, this method can accurately predict the position of both backbone and side-chain atoms. Moreover, it predicts the structure of the entire variable domain, not just that of the CDRs. Chapter 3 describes the ImmuneBuilder method in detail.

With next-generation sequencing studies routinely generating millions of antibody sequences, effectively searching this data has become a challenge. In Chapter 4, we present KA-Search [4] a tool that allows for the rapid search of billions of antibody sequences by sequence identity across either the whole chain, the complementarity-determining regions, or a user defined antibody region. We show KA-Search in operation on the 2.4 billion antibody sequences available in the OAS database [5].

In addition to next-generation sequencing data, there is a vast amount of literature that focuses on a small number of antibodies. For these smaller scale experiments, the amount of available information for each individual antibody is much greater. To facilitate the retrieval of this information, we developed the Patent and Literature Antibody Database (PLAbDab) [6], an evolving reference set of functionally diverse, literature-annotated antibody sequences and structures. Chapter 5 describes how this database was generated and gives examples of how it could be used.

The final chapter draws conclusions and proposes future avenues of research.

Papers

I am an author on the following papers and preprints:

- First Author: “ABlooper: Fast accurate antibody CDR loop structure prediction with accuracy estimation”. In: *Bioinformatics* 38.7 (2022), pp. 1877–1880.
- First Author: “ImmuneBuilder: Deep-Learning models for predicting the structures of immune proteins.” In: *Communications Biology* 6.1 (2023), p. 575.
- First Author: “The Patent and Literature Antibody Database (PLAbDab): an evolving reference set of functionally diverse, literature-annotated antibody sequences and structures” In: *bioRxiv* (2023).
- Co-First Author: “Advances in computational structure-based antibody design”. In: *Current Opinion in Structural Biology* 74 (2022), p. 102379.
- Co-First Author: “KA-Search: Rapid and exhaustive sequence identity search of known antibodies”. In: *Scientific Reports* 13.1 (2023), p. 11612.
- Contributing Author: “Improved computational epitope profiling using structural models identifies a broader diversity of antibodies that bind the same epitope”. In: *bioRxiv* (2023).

Contents

1	Introduction	1
1.1	Chapter abstract	2
1.2	Antibodies	2
1.2.1	Antibodies in the immune system	2
1.2.2	Antibodies as therapeutics	3
1.2.3	Antibody regions and numbering	4
1.2.4	Antibody sequence diversity	6
1.2.5	Antibody structure	7
1.2.6	Next generation sequencing of BCR repertoires	9
1.2.7	TCRs and nanobodies	10
1.3	Machine Learning	11
1.3.1	Introduction	11
1.3.2	Deep learning	12
1.3.3	Inductive bias	13
1.3.4	SE(3)-Equivariant graph neural networks	13
1.4	Protein structure prediction	15
1.4.1	Introduction	15
1.4.2	Early methods	16
1.4.3	AlphaFold2	17
1.4.4	AlphaFold2 architecture	20
1.4.5	OtherFolds	22
1.5	Antibody structure prediction	23
1.5.1	Applications of antibody structure prediction	24
1.5.2	Homology modelling	25
1.5.3	Deep learning	26
1.6	Chapter Walkthrough	28
2	ABlooper	30
2.1	Chapter abstract	30
2.2	Background	31
2.3	Methods	32
2.3.1	Data	32

2.3.2	Deep learning	33
2.3.3	Loop relaxation	35
2.3.4	Benchmarked methods	36
2.4	Results	36
2.4.1	Using ABlooper to predict CDR loops on modelled antibody structures	36
2.4.2	Prediction diversity as a measure of prediction quality . . .	37
2.5	Discussion	39
3	ImmuneBuilder	42
3.1	Chapter abstract	42
3.2	Background	43
3.3	Methods	46
3.3.1	Data	46
3.3.2	Deep learning architecture	47
3.3.3	Model selection	49
3.3.4	Structural refinement	49
3.3.5	Benchmarked methods	50
3.4	Results	51
3.4.1	Accuracy of prediction	52
3.4.2	Heavy and light chain packing	53
3.4.3	Side chain and chemical surface accuracy	55
3.4.4	Physical plausibility and accurate stereochemistry	56
3.4.5	Computational cost	57
3.4.6	Error estimation	57
3.4.7	Sequence identity against prediction accuracy	58
3.4.8	NanoBodyBuilder2 results	58
3.4.9	TCRBuilder2 results	60
3.5	Discussion	62
4	KA-Search	67
4.1	Chapter abstract	67
4.2	Background	68
4.3	Method	73
4.3.1	Data preprocessing	73
4.3.2	Identity calculation	75
4.3.3	Sensitivity and speed comparison	76
4.4	Results	77
4.4.1	Computational speed of KA-Search	77
4.4.2	Comparison with other common sequence identity search tools	78
4.4.3	Immune repertoire mining with the COVOX-253 antibody .	81
4.5	Discussion	83

5	PLAbDab	86
5.1	Chapter abstract	86
5.2	Background	87
5.3	Methods	88
5.3.1	Collecting unpaired antibody sequences	88
5.3.2	Creating paired antibody sequences from the unpaired data	88
5.3.3	Searching PLaBdab	90
5.4	Results	90
5.4.1	Database statistics	90
5.4.2	Searching PLaBdab	92
5.4.3	Using PLaBdab to Generate New Datasets	96
5.5	Discussion	96
6	Conclusions and Future Work	98
6.1	Conclusions	98
6.2	Future work	99
	References	102
	Appendices	
A	Machine learning datasets	118
A.1	ABlooper data	118
A.1.1	Rosetta Antibody Benchmark	118
A.1.2	SAbDab Latest Structures	118
A.1.3	Train and validation	119

1

Introduction

Contents

1.1	Chapter abstract	2
1.2	Antibodies	2
1.2.1	Antibodies in the immune system	2
1.2.2	Antibodies as therapeutics	3
1.2.3	Antibody regions and numbering	4
1.2.4	Antibody sequence diversity	6
1.2.5	Antibody structure	7
1.2.6	Next generation sequencing of BCR repertoires	9
1.2.7	TCRs and nanobodies	10
1.3	Machine Learning	11
1.3.1	Introduction	11
1.3.2	Deep learning	12
1.3.3	Inductive bias	13
1.3.4	SE(3)-Equivariant graph neural networks	13
1.4	Protein structure prediction	15
1.4.1	Introduction	15
1.4.2	Early methods	16
1.4.3	AlphaFold2	17
1.4.4	AlphaFold2 architecture	20
1.4.5	OtherFolds	22
1.5	Antibody structure prediction	23
1.5.1	Applications of antibody structure prediction	24
1.5.2	Homology modelling	25
1.5.3	Deep learning	26
1.6	Chapter Walkthrough	28

1.1 Chapter abstract

In this chapter, I provide the background needed to understand the research presented in this thesis. I begin by giving a brief overview of antibodies, a class of proteins crucial to the adaptive immune response in humans. I discuss their role in the immune system, the factors driving their diversity, both in sequence and structure, and how this determines their function.

Next, I introduce machine learning with a special focus on equivariant deep learning and its application to predicting protein structures. I provide an overview of the AlphaFold2 [3] architecture and describe how different architectural choices were made. Finally I describe the field of antibody structure prediction, and how deep learning has revolutionised the field in recent years.

1.2 Antibodies

1.2.1 Antibodies in the immune system

Antibodies are globular proteins produced by B-cells that play a critical role in the adaptive immune response of jawed vertebrates. They recognise and neutralise pathogenic organisms, or antigens, binding to them with high affinity and specificity [7]. Antibodies achieve their function by being able to exploit an immense diversity of binding sites, enabling them to attain a high level of complementarity with almost any antigen. They then induce downstream effector functions, including pathogen neutralisation and activation of other immune system components [8].

The diversity of antibody sequences is immense, with a theoretical sequence space of up to 10^{18} [9], suggesting the existence of at least one high-affinity binding partner for virtually any antigen. Antibodies are presented on circulating B-cells, each of which generates and presents a single type of antibody, commonly known as B-cell receptors (BCRs). The diversity of circulating antibodies is constrained by the number of circulating B-cells, which has been estimated to be around 10^9

[10]. As a result, antibodies in the bloodstream may initially lack optimal binding capabilities to a novel pathogen.

When a BCR binds to the antigen, it drives its parent B-cell to the germinal centre of the lymph nodes. Here, the DNA sequence of the B-cell undergoes mutations, altering the amino acids that constitute the BCR binding site. Mutations that enhance the affinity of the BCR towards its cognate antigen are positively selected, increasing the specificity of the BCR population within the germinal centre. This process is known as ‘affinity maturation’, and results in the secretion of highly specific antibodies of diverse isotypes and effector functions.

The optimisation of antibodies for pathogen neutralisation spans several days, accounting for the period between becoming sick and the beginning of recovery [11]. After neutralisation, B-cells encoding the optimised antibodies are retained at reduced concentrations as ‘memory B-cells’. In the event of reinfection by the same or a similar pathogen, these memory B-cells are reactivated, secreting optimised antibodies. Vaccination strategies capitalise on this feature, promoting immunological memory for a specific set of proteins presented on a pathogen that can be quickly recalled upon reinfection [12].

1.2.2 Antibodies as therapeutics

There are currently over a hundred FDA-approved antibody therapeutics [13], with an estimated combined market value of over \$200 billion US [14]. The ability of antibodies to bind almost any antigen with high affinity and specificity make them attractive candidates for therapeutic development [15]. Antibodies are currently used and actively being developed to treat a wide selection of diseases [13]. For example, since the start of 2022, antibody therapeutics have been approved by the FDA to treat RSV in infants (Nirsevimab [16]), multiple sclerosis (Ublituximab [17]), and malignant melanoma (Relatlimab [18]).

Other than their binding abilities, an aspect of antibodies that makes them attractive as therapeutics is their ability to elicit effector functions [14]. Antibodies can activate the immune system through their constant region, triggering processes

such as antibody-dependent cellular cytotoxicity (ADCC), or antibody-dependent cellular phagocytosis (ADCP) [19]. These mechanisms contribute to the elimination of target cells in a range of diseases, including cancer [20] and HIV [21].

The modular structure of antibodies also allows for bioengineering approaches to enhance their efficacy and improve their pharmacokinetic profiles [22]. Techniques such as humanisation [23, 24] and affinity maturation [25] can be used to decrease immunogenicity and increase binding affinity. Additionally, the development of bispecific [26] antibodies and antibody-drug conjugates [27] broadens the therapeutic and diagnostic potential of antibodies, allowing simultaneous engagement of multiple targets or delivering potent drugs to specific cells, respectively.

The well-established manufacturing and regulatory pathways for therapeutic antibodies also contribute to their prominence as therapeutics [28]. With the increasing knowledge about antibody structure and function, as well as advances in antibody engineering and production techniques, the development of therapeutic antibodies will likely continue to be a major focus in the field of drug discovery [14, 29].

1.2.3 Antibody regions and numbering

A typical human antibody consists of two sets of heavy and light chains that form two antigen binding regions (Fab) and a highly conserved region (Fc). The effector function of the antibody is determined by the Fc region, which allows for recognition of different antibody isotypes by their receptors. Each Fab is composed of a conserved and variable region (Fv), where the variable region provides the diversity in sequence and structure necessary to bind a wide range of antigens. A pictorial representation of a typical human antibody structure can be seen in Figure 1.1.

The Fv region is formed by two immunoglobulin variable domains, one from the heavy chain (VH) and one from the light (VL) [30]. Each of the VH and VL domains contain three hypervariable loops, also known as complementarity-determining regions (CDRs) [31]. These CDRs are responsible for directly interacting with the antigen and determine the antibody's specificity. The CDRs are designated as CDR1, CDR2, and CDR3 and are located at the tip of the VH and VL domains.

CDR3 loops, in particular CDR-H3, are the most diverse and often the largest contributors to binding [32, 33]. They are formed by the most variable regions in the antibody gene sequences, allowing them to recognise a wide variety of antigens. The CDR1 and CDR2 loops also contribute to antigen binding but are generally less variable than CDR3 [34, 35]. Fv amino acids that are not part of the CDR loops are referred to as the framework.

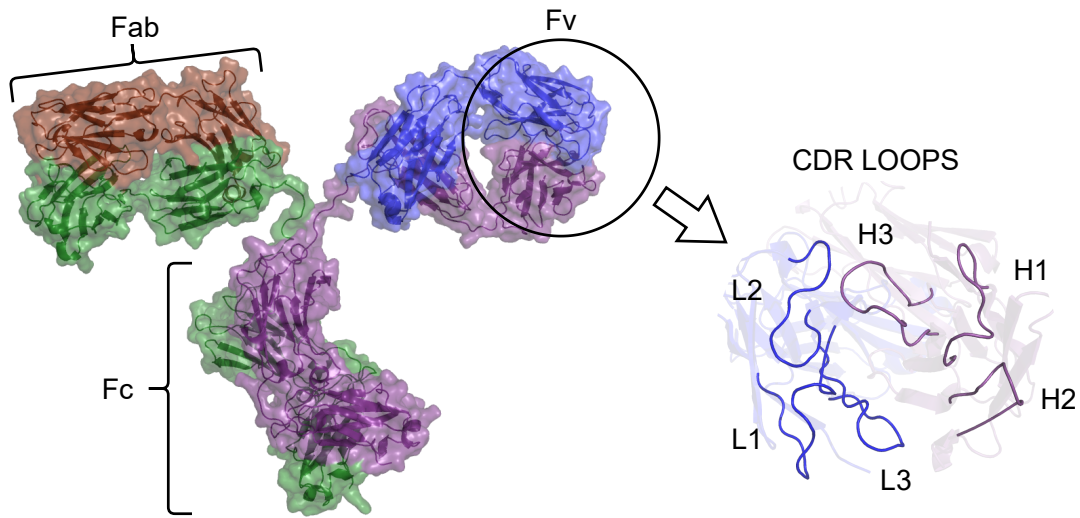


Figure 1.1: Schematic of a standard antibody, showing its two heavy chains (purple and green) and light chains (brown and blue). Antibodies contain two antigen binding regions (Fab) and one crystallisable domain (Fc). The area responsible for binding in the Fab is called the variable region (Fv), with most of the variability required for binding concentrated in six loops known as the complementarity determining regions (CDRs). The part of the Fv that does not belong to the CDRs is called the framework. On the right side of the figure, the structure of the Fv region is shown, with the CDR loops highlighted and labelled. PDB IDs: 1IGT (left), 15C8 (right). Figure reproduced from [36].

Antibody numbering schemes have arisen in the field of immunoinformatics as a systematic way to compare antibody sequences without relying on sequence alignments [37]. The underlying principle of antibody numbering schemes lies in assigning unique identifiers or positions to each residue of the VH and VL. These positions can help identify and compare regions across different antibody sequences, such as the CDRs. Antibody numbering schemes often involve defining a set of standardised reference antibody sequences against which other sequences can be

compared. Various numbering systems exist, such as the Kabat, Chothia, and IMGT numbering schemes, each with their own specific strengths and conventions [30, 38, 39]. In this thesis, antibody numbering is used to identify what residues belong to the CDR loops, with the choice of scheme having little impact on the presented results.

1.2.4 Antibody sequence diversity

The diversity in antibody sequence space is largely driven by two mechanisms: V(D)J recombination and somatic hypermutations. V(D)J recombination is a somatic recombination process that takes place during B-cell development, and involves the recombination of variable (V), diversity (D), and joining (J) gene segments. This process generates a unique V(D)J exon that encodes for the antibody VH, and a similar but less complex assembly occurs to complete the VL [7]. By encoding the Fv in terms of combinations of gene segments, the body can access a wider sequence space while using a significantly smaller portion of the genome. In humans, there are thought to be 57 V genes, 6 J segments, and 23 D segments for the heavy chains, 50 V genes, and 5 J segments for kappa light chains, and 50 V genes, and 7 J segments for lambda light chains [40]. The number of possible combinations of these genes alone suggests an antibody sequence diversity of around 4.5 million. Additional modifications to the sequence can occur when stitching these genes together, such as insertions or deletions of nucleotides at the V-J and J-D junctions, which further increase the coverage of antibody sequence space [41]. For other mechanisms that lead to an increased sequence diversity during V(D)J recombination, see [9].

The second major driver for antibody sequence diversity are somatic hypermutations (SHM). SHMs are point mutations to the the variable region that occur with high frequency during affinity maturation in the germinal centre. The overall goal of affinity maturation is to produce antibodies that bind strongly to an antigen. SHMs by themselves do not distinguish between favourable and unfavourable mutations, but B-cells with favourable mutations are positively selected in the germinal centre during affinity maturation [42]. The majority of mutations tend to accumulate in the CDR loops, as these typically have the largest effect on

binding. Evolutionary selection has facilitated this process by enriching the CDRs for codons with activation-induced cytidine deaminase (AID) hot spots that result in a higher frequency of mutations [43–45].

1.2.5 Antibody structure

Antibodies have a characteristic Y-shaped architecture, built from two identical heavy chains and two identical light chains, interconnected by disulfide bridges (see Figure 1.1A). Each heavy and light chain has a variable domain (VH or VL) at the N-terminus, which contributes to the formation of the Fv, and at least one constant domain at the C-terminus. The variable domains exhibit an immunoglobulin fold, a highly stable structural motif consisting of two layers of anti-parallel beta sheets connected by a series of loops [46]. Six of these loops, three from each domain, form the CDRs and are the main contributors to antibody-antigen binding. The VH and VL domains associate in a non-covalent manner to bring the CDR loops together, thus forming the antigen binding site [47].

Five of the six CDR loops (L1, L2, L3, H1 and H2) are often observed in a limited number of conformations known as canonical forms [48]. Canonical forms refer to the concept that sets of CDR loop sequences consistently adopt similar three-dimensional structures, primarily driven by their sequence composition and length. As more antibody structural data has become available, there have been a number of studies that have attempted to classify canonical forms [48–51]. As a result, it is possible to predict the canonical class of a CDR loop with a high level of accuracy based solely on its sequence [52].

The third CDR loop of the heavy chain (H3) exhibits the most structural diversity, with a wide range of potential conformations observable even among antibodies with similar sequence [34]. The sequence and structural diversity of CDR-H3 plays a critical role in antigen binding, with alterations in loop length and sequence often directly impacting the antibody’s antigen affinity. This diversity has also made CDR-H3 the hardest part of the antibody to model [53].

With three CDRs on each domain, the relative orientation between the VH and VL can significantly affect the structure of the binding site [54]. The relative orientation between the VH and VL has been fully characterised using five angles and a distance [55], as shown in Figure 1.2. The VH-VL packing angle is determined by interacting residues on the interface, most of which are on the framework [56]. Due to this, packing angles can often be predicted from the antibody sequence, although small errors can have a significant impact on the final structure of the binding site [57].

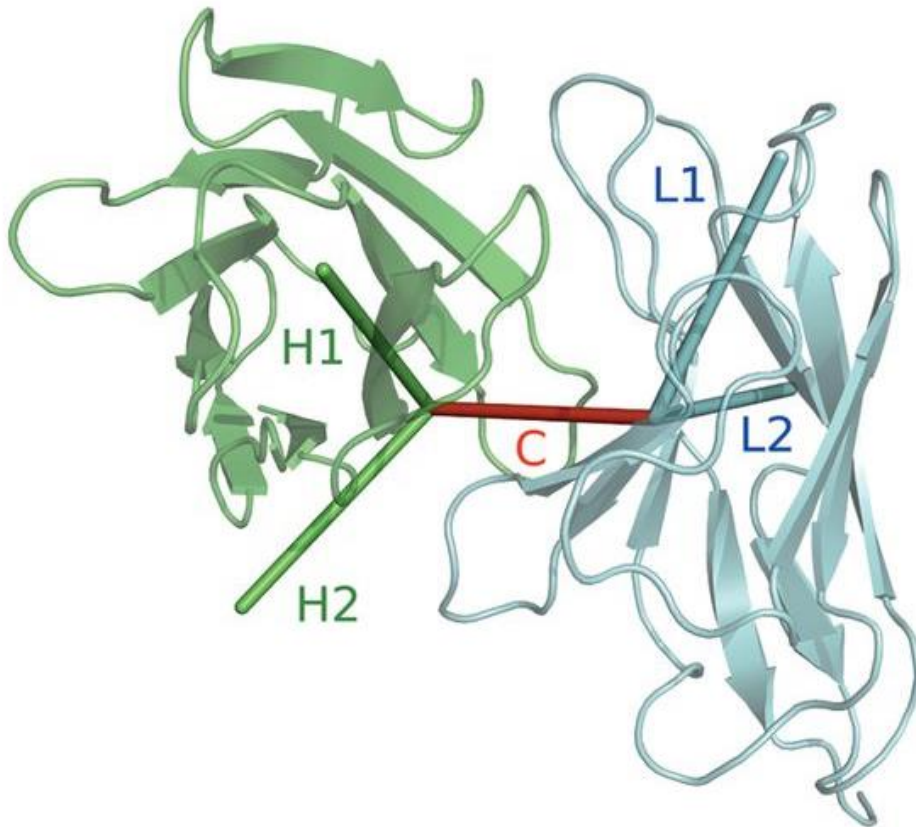


Figure 1.2: Antibody structure with the five vectors used to calculate the 6 values used to define heavy and light chain packing. The distance d_c is defined as the length of the vector C. The HL angle is defined as the torsion angle between the H1 and L1 vectors measured about C. The HC1, HC2, LC1 and LC2 angles are the angles between the H1, H2, L1 and L2 vectors and the vector C. Figure reproduced from [55].

1.2.6 Next generation sequencing of BCR repertoires

While structural information can provide valuable insights into the function of individual antibodies, next-generation sequencing (NGS) of B-cell receptors allows for studies of the antibody repertoire in a given individual. NGS of B-cell receptor repertoires has revolutionised our understanding of the adaptive immune system [58]. It enables the sequencing and analysis of millions of antibody chains, providing valuable insights into how the immune system responds to different antigens across various organisms and disease stages [59–61]. The continuous improvement of high-throughput sequencing methods and their widespread adoption in research labs contribute to the rapid increase in the volume and diversity of publicly available antibody sequence data [10, 62].

NGS sequencing generates large amounts of data at the nucleotide level, which requires meticulous cleaning and processing [63, 64]. Efforts to compile this data into a unified format include the Observed Antibody Space (OAS) [65] and iReceptor [66]. These large repositories, housing several billion antibody sequences, are being used for machine learning [67–69], vaccine development [70, 71], and drug discovery [72, 73]. Due to limitations in the experimental methods used for NGS, most studies only sequence the VH and VL domains, home to the binding site, instead of the full antibody sequence [65, 66].

Although NGS studies have mainly focused on the exploration of unpaired heavy chain BCR repertoires, information about the native pairing of VH and VL can provide deeper insights into antibody repertoire diversity and dynamics, and is required to build a structural representation of the Fv [74, 75]. The decision to avoid paired sequencing of natively associated VH and VL domains has been primarily due to the complexity of high-throughput experimental setups needed to identify VH and VL mRNA molecules originating from the same B-cell [76].

The recent 10xGenomics chip offers a streamlined solution for paired VH-VL sequencing, although it is significantly lower-throughput than single chain methods and more expensive [77]. Studies based on paired sequencing typically generate tens of thousands of BCR sequences [78, 79], although recent studies

have managed to generate over a million [80]. As a result, there is far less paired VH-VL antibody data available [5].

1.2.7 TCRs and nanobodies

To effectively identify and neutralise a wide range of pathogens, the adaptive immune system in humans does not solely rely on B-cells and antibodies. For example, T-cell receptors (TCRs) are antigen-specific proteins that have evolved to bind fragments of intracellular proteins presented on the cell surface by the major histocompatibility complex (MHC). Due to their ability to identify what is occurring inside a cell, TCRs have also become proteins of interest for therapeutic development, with several TCR drugs in clinical trials [81]. Nanobodies, single-domain antibodies naturally occurring in organisms such as camelids and sharks, have also received significant interest as therapeutics, with a recently approved nanobody drug and a number undergoing clinical trials [82].

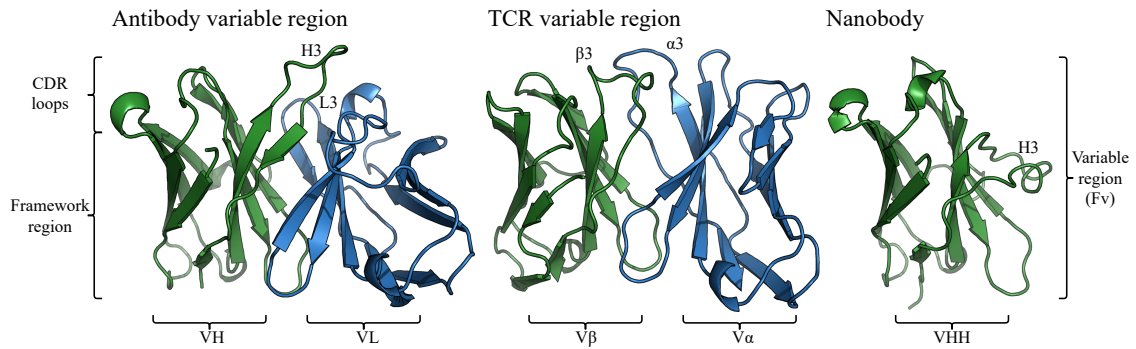


Figure 1.3: Structural representation of an antibody variable domain (PDB code 1GIG), a TCR variable domain (PDB code 7SU9) and a nanobody (PDB code 4LAJ) with labelled regions. The heavy and beta chains are shown in green while the light and alpha chains are blue. As these structures show, antibodies and TCRs are structurally similar and the nanobody is also similar to an individual chain of an antibody or TCR. However, their CDR loops occupy distinct areas of structural space.

Similar to antibodies, nanobodies and TCRs are also composed of immunoglobulin (Ig) domains, with the binding site sitting across two Ig domains in the case of TCRs ($V\alpha$ and $V\beta$), or located at the tip of one Ig domain (VHH) in the case of nanobodies. The binding site of TCRs is also concentrated in six CDR loops, with three on each of the two Ig domains. In nanobodies, the binding site is concentrated

in only three CDR loops on its single Ig domain. These CDR loops show variable length, composition, and structure, with the most variable being CDR3 for both nanobodies and TCRs. Similar to antibodies, CDR3 loops also tend to be the largest contributors to binding [83, 84]. The structures of an antibody variable domain, a TCR variable domain, and a nanobody are depicted in Figure 1.3.

Despite the similarities in the global structure of antibodies, TCRs and nanobodies, their binding sites are known to have distinct properties and their CDRs have different length distributions as well as occupying distinct areas of structural space [83–85].

1.3 Machine Learning

1.3.1 Introduction

A large part of this thesis focuses on the application of machine learning to predict the structure of antibodies from their sequence. Machine learning is a sub-field of artificial intelligence that involves the development of statistical models that enable computer systems to learn from and make predictions based on data, without being explicitly programmed to do so. The machine learning process involves optimising the parameters of a model (often called ‘training’) to identify patterns and relationships within data. Once a model has been trained, it can be used to make predictions on new, unseen data.

Machine learning models are often classified into supervised, unsupervised, or reinforcement learning based on the training strategy used to build them [86]. In supervised learning, the model is trained on labelled data, where each data point has a known input and output. It then learns to generalise the relationship between the input and output variables to make predictions on new, unseen data. For unsupervised learning, the model is trained on unlabelled data, where the input data does not have a known output. It then learns to find patterns and structure within the data to create groups or clusters of similar data points. In reinforcement learning there is not a fixed data set, the model learns through trial and error by interacting with the environment. It receives feedback in the form

of rewards or punishments based on its actions, and it learns to make decisions that maximise the reward over time. These three categories of machine learning are not fully disjoint and are often used in combination to build more complex models that can handle a wider range of problems.

1.3.2 Deep learning

Traditional machine learning techniques have often required extensive domain expertise to engineer feature extractors. The role of these extractors is to transform raw data into an appropriate feature vector, from which the model can learn patterns in the data. Deep learning has emerged as a branch of machine learning in which the model is tasked with learning from raw data. This is often achieved by using neural networks with multiple layers [87].

Neural networks are a class of machine learning model where adjustable parameters are held in matrices known as layers. The initial data is received by the input layer, while the output layer generates the final result. Any intermediate layers that process and transform data are referred to as hidden layers. Neural networks are trained by adjusting the parameters stored in their layers to minimise a defined loss function on a specified dataset, employing various optimisation algorithms such as Adam [88]. In theory, the loss function could be any function that measures the difference between the actual data and the model's predictions, although in practice, most commonly used optimizers require it to also be differentiable [89].

Deep neural networks automatically learn hierarchical representations of the data by successively extracting increasingly complex features at each layer. The initial layers of the network learn simple and local patterns, while subsequent layers learn more abstract and high-level representations. This hierarchical feature extraction allows deep neural networks to capture intricate dependencies and subtle relationships within the data, leading to superior performance in various tasks [90, 91].

1.3.3 Inductive bias

In theory, neural networks with at least one hidden layer have the ability to approximate any continuous function with arbitrary accuracy, given a sufficient number of hidden units [92, 93]. This property has contributed to the widespread success of neural networks. However, in practical applications, machine learning engineers often possess domain knowledge about the data, which can be utilised to enhance the model’s performance. This knowledge, known as inductive bias, constrains the set of possible functions that the model can learn, guiding its search towards functions that are more likely to generalise well. One way to incorporate inductive bias into the model is via a term in the loss function. Weight decay, for example, is a term added to the loss function that biases the model towards using fewer parameters, assuming that a simpler model is more likely to generalise [94].

Another method for incorporating inductive bias into a model is to design its architecture so it will respect certain symmetries observed in the data. An example of this are convolutional neural networks (CNNs), commonly used in computer vision [91, 95]. CNNs have been designed to be equivariant to 2D translations, a symmetry present in most image data. By employing sliding filters, CNNs can recognise objects regardless of their position within an image. The design of CNNs ensures that the model is not aware of where the pixels it is analysing are globally placed on the image. Instead, it focuses on small neighbouring regions of pixels at a time, thereby also incorporating a locality bias. This bias comes from the assumption that neighbouring pixels provide more relevant information compared to distant pixels. By incorporating the inductive bias of translational equivariance into the model, CNNs effectively leverage the local context for tasks like object recognition.

1.3.4 SE(3)-Equivariant graph neural networks

Deep learning models designed to exploit certain symmetries in the data are often referred to as being equivariant to that symmetry. For example, CNNs are equivariant to 2D translations. More rigorously, a function (f) is said to be equivariant to a group (G) if it commutes with the group action (g). This means,

that any transformation belonging to G performed on the input of f yields an equivalent result to the transformation performed on the output. The function f is by definition equivariant to the group G if it satisfies the following equation:

$$g * f(x) = f(g * x) \quad \forall g \in G \quad (1.1)$$

Graph neural networks (GNNs) [96] are a type of deep learning architecture designed to deal with graph structured data, where nodes represent entities and edges represent relationships between them. GNNs are by design equivariant to permutations, as the order of nodes in the graph should not affect the output of the model. GNNs achieve permutation equivariance by employing aggregation functions, such as the sum or mean, that aggregate information from neighbouring nodes. These aggregation functions treat neighbouring nodes equally, regardless of their specific order. GNNs have been successful in various applications, such as molecular chemistry, social network analysis, and recommendation systems [97].

In structural bioinformatics, atoms or molecules and their interactions can be represented as nodes and edges on a graph. These graphs are typically conceptualised as point clouds in three-dimensional space. For most molecular data, the energy of a system does usually not depend on the molecule’s absolute position or orientation in space but rather, the relative position and orientation between molecules. This relative position and orientation will determine if and how pairs of molecules interact. It is therefore of interest to develop models capable of learning relationships in structural data, regardless of the global reference frame.

The group that describes rotations and translations in 3D space is called SE(3). A function will demonstrate SE(3)-equivariance if executing a rigid transformation on the data before inputting it into the function yields the same result as performing the transformation on the output. Due to the ability of GNNs to consider relative interactions between nodes, they present themselves as the ideal framework for building SE(3)-equivariant neural networks [98].

Several deep learning architectures have been proposed to achieve SE(3)-equivariance for point clouds in recent years. Perhaps the first to be developed were

Tensor Field Networks (TFN), where they describe geometric features in terms of spherical harmonics and then aggregate them using combinations of tensor products weighted by learnt parameters [99, 100]. It has been found that combining TFNs with the transformer architecture [101] can lead to improved accuracy at the cost of computational speed [102]. More general architectures (such as EMLP [103]) have been designed to achieve equivariance for any Lie group (group whose elements form a continuous and differentiable manifold). These architectures are all very general, and allow you to have higher-ranking geometrical tensors (such as matrices) as node features [104]. However, describing features in terms of spherical harmonics or Lie group actions [103] make these methods computationally expensive and hence limit the number of possible learnable parameters that can be used in practice.

Less universal but more computationally efficient approaches to SE(3) equivariance include E(n) equivariant graph neural networks (E(n)-EGNNs) [105], vector neurons [106], the geometric vector perceptron [107, 108], and the AlphaFold2 [3] structure module. E(n)-EGNNs only allow nodes to have vector and scalar features and are equivariant to translations, rotations and reflections. In E(n)-EGNNs, equivariance is achieved by computing coordinate updates as a weighted sum of vectors between nodes. The E(n)-EGNN architecture is explained in depth in Chapter 2. Vector neurons and the geometric vector perceptron both use modified linear layers and activation functions to preserve rotation equivariance. Translation equivariance is achieved by using these new perceptrons to build a GNN that acts on relative coordinates and features. The AlphaFold2 architecture is described in Section 1.4.4

1.4 Protein structure prediction

1.4.1 Introduction

Antibodies represent just one important class of the wide variety of proteins in the human body. In fact, proteins perform the majority of vital functions necessary for life. Proteins are polymers, extended molecule chains composed of a sequence of interconnected smaller molecules. The vast majority of natural proteins are

constructed from combinations of just twenty different types of molecules, known as amino acids. Through interactions between amino acids, proteins fold into three-dimensional structures. For most proteins, it is thought that the specific arrangement of amino acids in the chain fully determines the final folded structure [109].

Understanding the three-dimensional structure of a protein is crucial for comprehending its function. For instance, the structure of a protein will determine which residues can interact with other molecules in its environment and how such interactions will occur. However, experimentally determining the structure of a protein is time-consuming and expensive [110]. Computational methods to predict the structure of proteins have proven to be of great value for better understanding the function of a protein without having to solve it experimentally [111].

1.4.2 Early methods

Proteins possessing highly identical amino acid sequences often fold into similar three-dimensional structures. Homology modelling is a strategy by which protein models for a given sequence are built from already solved structures with a similar sequence [112]. These methods have proven to be highly successful when the structure of a homologous protein exists [113–116], but struggle when there are no structures with a similar sequence available [117]. For instance, in the field of antibodies, homology modelling methods have demonstrated sub-angstrom accuracy when predicting the framework region [118] (see Section 1.2.3).

Traditionally, when no structures of homologs were available, researchers predicted the protein structure by modelling interactions between pairs of amino acids. These methods include molecular dynamics force fields, which attempt to model the underlying physical interactions [119, 120], and strategies based on statistical potentials derived from previously solved structures [121, 122]. Although these methods struggle to accurately model structures from scratch, they are still used today in conjunction with other techniques.

One of the key findings that has led to progress in the field of protein structure prediction comes from the idea of co-evolution [123]. When two amino acids are

close in three-dimensional space, one of them mutating can change how they interact, which may affect the proteins function, stability, or ability to fold. To maintain this mutation, a corresponding mutation in the other amino acid, which restores the interaction, is necessary. If an interaction is crucial to a protein’s function or stability, it is the interaction between co-evolving amino acid pairs that is preserved, rather than their identity. In practice, this means that if two amino acids are observed to have mutated together throughout evolution, they are likely to be interacting and are hence close in three-dimensional space [124].

Statistical methods can be employed on large sets of protein sequences from various organisms to infer evolutionary couplings between residues. Early co-evolution based methods used multiple sequence alignments (MSAs) to predict if two residues are in contact, and then used these contact maps to guide physics-based methods [125, 126]. Deep learning was first introduced in the field of protein structure prediction through contact prediction [127, 128]. RaptorX [129] and the original version of AlphaFold [130] used features derived from MSAs to predict distances between residues. This approach was then extended by TrRosetta [131] to also predict relative orientations.

1.4.3 AlphaFold2

At the 14th round of the Critical Assessment of Structure Prediction (CASP14) [132], DeepMind unveiled AlphaFold2 [3], a deep neural network capable of predicting a vast number of protein structures with remarkable accuracy. The network predicts structures with an average Global Distance Test (GDT) score close to 90 for individual domains, a level of accuracy considered comparable to experimental structures [132]. Upon its release, AlphaFold2 was significantly more advanced than any other available method for protein structure prediction (see Figure 1.4). Even two years later at CASP15, AlphaFold2 was shown to still be one of the most accurate structure prediction methods available [133].

The success of AlphaFold2 can be attributed to the accumulation of small improvements from a variety of well-executed ideas (see ablation studies in [3]).

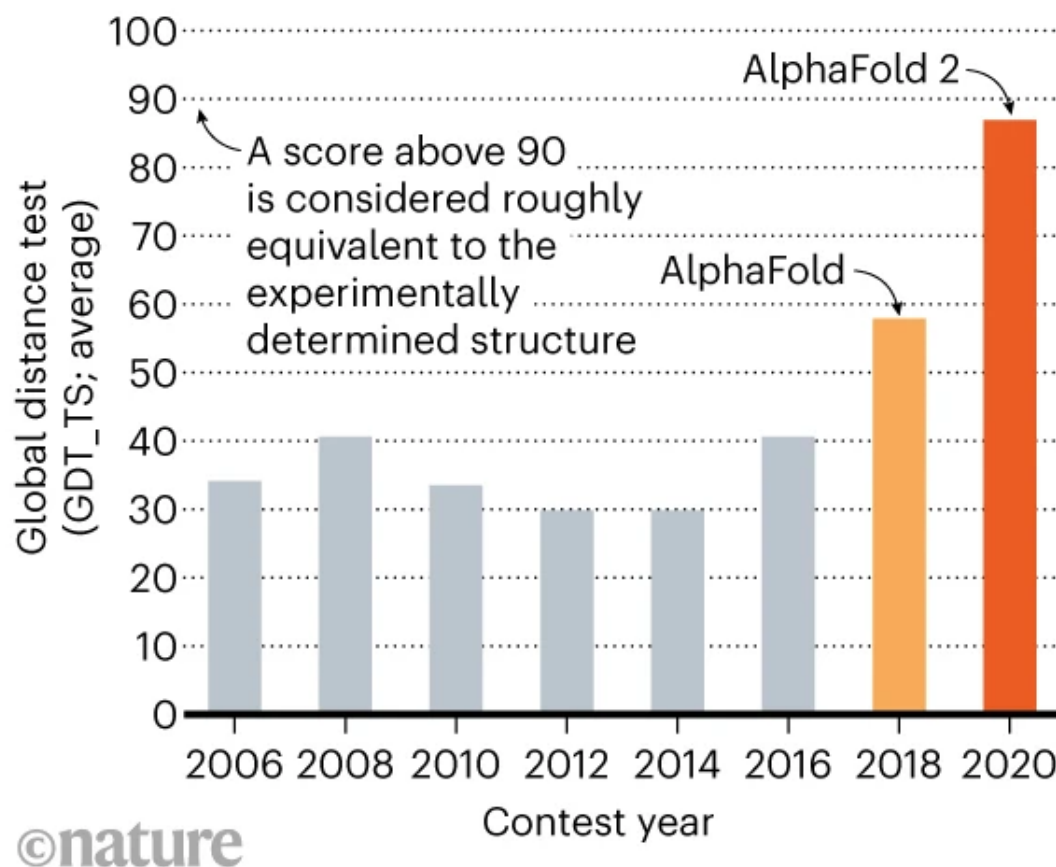


Figure 1.4: DeepMind’s AlphaFold2 algorithm significantly outperforms other teams at the CASP14 protein structure prediction contest. The GDT_TS score is a way to measure the similarity between a predicted protein structure and the actual experimental structure. This is done by examining the proximity of the carbon alpha atoms in the predicted model to those in the real protein structure. The GDT_TS is scored on a scale of 0 to 100; a higher score indicates greater similarity between the structures. A score of 100 signifies that the structures are identical. Figure reproduced with permission from [134].

However, the majority of these improvements can be categorised as falling under one of two general concepts: the construction of an end-to-end model and the incorporation of inductive bias into the model.

Diverging from all previous protein structure prediction models, where the input features were precomputed by extracting co-evolutionary information from an MSA using statistical methods, AlphaFold2 takes the MSA directly as input. Additionally, instead of predicting inter-residue distances and orientations, AlphaFold2 directly predicts the three-dimensional structure of the protein. Designing the model to be

end-to-end allows it to directly identify important correlations in the data, rather than learning relationships between aspects considered important by the model’s designer. This progress parallels the advancement introduced by CNNs for image analysis, wherein a significant boost in accuracy was realised when researchers transitioned to using learned kernels instead of precomputed features.

Knowledge about the expected appearance and behaviour of a protein structure is built into the AlphaFold2 model. For instance, the architecture of AlphaFold2 is fully SE(3)-equivariant, enabling it to learn directly from the protein structure regardless of its spatial position and orientation. Although several SE(3)-equivariant methods for structure prediction predated AlphaFold2, none managed to leverage this as effectively [135, 136]. In AlphaFold2, residues are treated as rigid bodies with a defined position and orientation, and the side chains’ degrees of freedom are defined in terms of their torsion angles (see Figure 1.5e). This builds into the model the knowledge that distances and angles between bonded atoms in a residue are highly conserved. The primary loss function used, the frame-aligned point error (FAPE), emphasises accurate positioning of nearby residues in relation to each residue’s orientation, simulating what residue-residue interactions would look like (see Figure 1.5f).

As the FAPE loss function played a key role in the success of the AlphaFold2 models and is frequently referred to in Chapter 3, I will outline the equation used to calculate it below. Given a set of true and predicted residue frames (T) along with their corresponding coordinates (x), the FAPE loss can be computed as:

$$\text{FAPE_loss} = \frac{1}{ZN_{res}^2} \sum_{i=1}^{N_{res}} \sum_{j=1}^{N_{res}} \min \left(\left\| T_i^{p-1} \circ \mathbf{x}_j^p - T_i^{t-1} \circ \mathbf{x}_j^t \right\|, d_{clamp} \right) \quad (1.2)$$

Where, N_{res} represents the total number of residues, and Z along with d_{clamp} are constants set to $10\text{\AA}$ in the original AlphaFold2 paper. Essentially, the function compares the positions of residues in the local reference frame of each residue. By

clamping errors larger than 10Å, it also forces the model to prioritise the correct placement of neighbouring residues.

Several notable enhancements do not specifically align with the overarching concepts of an end-to-end model or the incorporation of inductive bias. For example, recycling, the concept of iteratively feeding predictions back into the model as templates, was shown to significantly boost accuracy. Another example is self-distillation, where a model trained on a blend of real and predicted data was found to surpass a model trained solely on real data.

1.4.4 AlphaFold2 architecture

The architecture of AlphaFold2 consists of two main modules: the Evoformer and the Structure Module. The Evoformer converts evolutionary couplings from large sets of similar sequences into information-rich embeddings (see Figure 1.5a). These embeddings are then used by the Structure Module to predict the three-dimensional structure of the protein (see Figure 1.5d).

The majority of the trainable parameters and computations in the AlphaFold2 model are part of the Evoformer. The Evoformer accepts an MSA as input and can optionally incorporate pairwise features extracted from a template structure. It then outputs both a pairwise and a single sequence representation. Its architecture can be divided into two sections. The first section updates the MSA features by running them through attention layers over the sequence and MSA-depth dimensions. Pairwise features are integrated at this stage via a bias term to the attention matrix. This alternating between column and row attention enables the model to learn from other residues in the sequence and variations in residues at the same position, similar to the approach used for the MSATransformer [137]. In the second section, the pairwise features are updated using triangular attention blocks (see Figure 1.5c) that bias the network towards respecting the triangle inequality between edges in the graph (see Figure 1.5b). The Evoformer is built from 48 blocks each of which alternates between updating the MSA and pairwise representations.

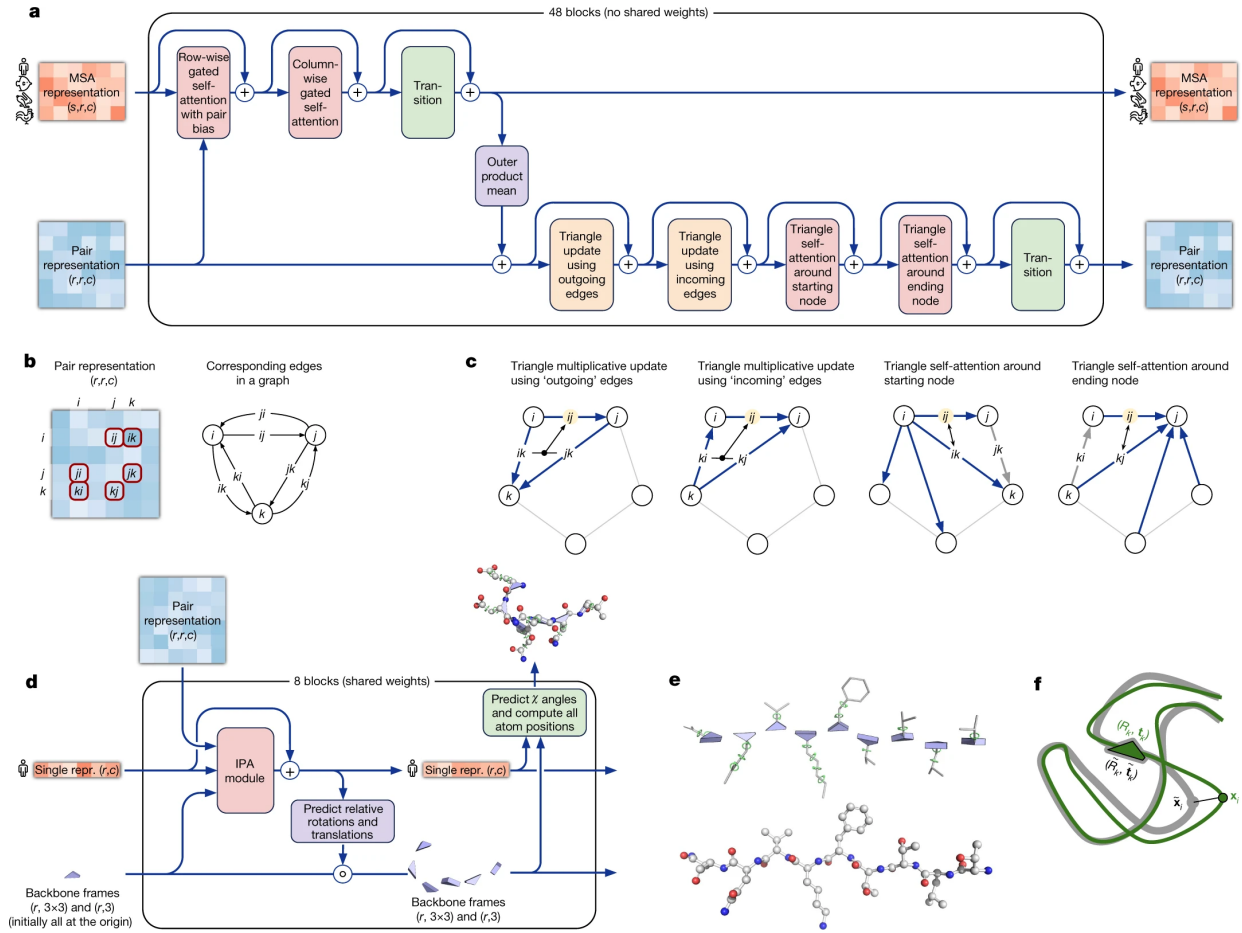


Figure 1.5: **a** Evoformer block. Arrows show the information flow. The shape of the arrays is shown in parentheses. **b** The pair representation interpreted as directed edges in a graph. **c** Triangle multiplicative update and triangle self-attention. The circles represent residues. Entries in the pair representation are illustrated as directed edges and in each diagram, the edge being updated is ij . **d** Structure module including Invariant point attention (IPA) module. The single representation is a copy of the first row of the MSA representation. **e** Representation of each residue as one free-floating rigid body for the backbone (blue triangles) and dihedral angles for the side chains (green circles). The corresponding atomic structure is shown below. **f** Frame aligned point error (FAPE). Green, predicted structure; grey, true structure; (R_k, t_k) , frames; x_i , atom positions. Figure reproduced from [3].

The Structure Module takes the pairwise and single representations generated by the Evoformer and uses them to predict the three dimensional structure of the protein. Each block within the Structure Module alternates between integrating information from the current structure and the pairwise representation into the single sequence representation, and iteratively updating the position and orientation of each residue. The module employs Invariant Point Attention (IPA) to update

the single sequence representation in a structurally aware manner. IPA is a type of attention designed to learn from a set of rigid bodies in a way that is invariant under global rotations and translations. The updated embeddings from IPA are subsequently used to predict updates to the position and orientation of each residue. Position updates are predicted as vectors in the local reference frame of the residue, while updates to orientation are predicted as quaternions. The single sequence representation is also employed to predict the dihedral angles of the side chain. This, in conjunction with the backbone reference frame, is used to calculate the position of all heavy atoms in a residue based on hard-coded rules (see Figure 1.5e). By building the structure in this way, AlphaFold2 ensures that the distances and angles between bonded atoms within a residue are physically accurate, reducing the likelihood of generating unphysical structures. To resolve any remaining structural violations and clashes, the structure predicated by the deep learning model is then minimised using the AMBER99SB [138] force field.

1.4.5 OtherFolds

Following the release of the AlphaFold2 paper and inference code, several research groups have strived to reproduce and improve its capabilities in a variety of ways [139–145]. A particularly significant extension to AlphaFold2 was developed by DeepMind itself: a model capable of predicting protein-protein complexes [146]. For this purpose, they introduced AlphaFold-Multimer, a variant of the original AlphaFold2, which bypasses the necessity for protein-protein docking methods [147], instead predicting the structure of the complex directly.

The majority of modifications to the AlphaFold2 architecture attempt to improve its speed or accuracy. The most time-consuming phase of creating a structure with AlphaFold2 is the generation of an MSA from large databases of protein sequences. AlphaFold2 uses a combination of JackHMMER [148] and HHBlits [149] to generate MSAs. However, this process can be substantially accelerated by using less exhaustive methods such as MMSeqs [150], which have been demonstrated to yield comparable results [151]. Several groups have attempted to entirely eliminate

the MSA generation step by substituting the Evoformer with a large language model pretrained on large sequence datasets [140–142]. By completely removing the MSA generation step, these methods can achieve a tenfold increase in speed compared to the original AlphaFold2. However, they have been shown to be less accurate than MSA-based methods [133]. In theory, omitting the MSA step could also help models to better generalise to proteins with sparse sequence information, but this approach has seen limited success.

There have also been several methods attempting to improve the accuracy of AlphaFold2. For instance, EquiFold [152] aims to enhance side chain prediction by treating each rigid group in the side chain as an independent entity capable of moving and interacting with other side chain rigid groups. This alteration enables the model to focus more on the accurate placement of side chains. However, it means that angles and distances between bonded atoms are no longer fixed, which increases the risk of generating unphysical structures. In RosettaFold2 [153], the authors explore which parts of the AlphaFold2 model are essential to its success and which components can be replaced by alternative architectures. They find that FAPE loss, recycling, and self-distillation are critical for accurately modelling protein structures, while most other architectural decisions can be replaced by alternatives, provided that the necessary symmetries are preserved. They then constructed a model with an architecture substantially different from AlphaFold2 but capable of matching its accuracy.

1.5 Antibody structure prediction

As with many proteins, the availability of antibody sequence data far outstrips the number of available structures [5, 65, 154–156]. However, structural information provides a more profound understanding than studies focused on sequences alone (see Section 1.5.1). Unfortunately, experimental structure determination is both time-consuming and expensive [110]. Computational modelling has emerged as a useful tool for easily obtaining accurate antibody structures from large volumes of sequence data [1, 2, 118]. Nevertheless, in order to facilitate structural analysis of datasets as

large as the Observed Antibody Space (OAS) [5, 65], the methods used to generate these structures would have to be orders of magnitude faster than AlphaFold2.

1.5.1 Applications of antibody structure prediction

Antibody structure prediction is already the first step in numerous computational analysis pipelines. By positioning residues within a three-dimensional context, predicted structures can offer valuable insights. For example, it aids in the identification of interacting residues, or in the discernment between whether a residue is exposed or buried [36].

The path from an antibody candidate to a therapeutic product is riddled with potential pitfalls. Unfavourable properties, such as self-association, polyspecificity, instability, high viscosity, immunogenicity, or poor expression, can render an antibody ineffective as a therapeutic agent [157]. An application of structural models that has shown great success is in predicting whether a candidate antibody will run into any of these developability issues [158, 159]. Predicted antibody structures facilitate this evaluation by enabling a comparative analysis of structural attributes, such as patches of high hydrophobicity or charge, in the antibody in question against antibodies that have successfully progressed past early stages of clinical trials. Using this comparative approach, we can predict and flag potential developability issues that the candidate antibody may encounter.

Another use of predicted antibody structures is for epitope binning. It has been shown that clusters of antibodies with a similar predicted structure across the CDR loops tend to target similar epitopes [160]. For antibodies that target the same antigen, clustering antibodies based on structural models can accurately predict which epitope they target [161]. Comparative analysis of the antibody binding sites of predicted structures has also been used to compare how structurally diverse mature and naive antibodies are [162], and to identify functional convergence for antibodies from different genes across the human population [75].

Generative deep learning models for antibody sequence design hold great promise in accelerating the drug discovery process [163, 164]. While most generative models

for antibody design are sequence-based [165–167], a few protein design methods have attempted to condition the generated sequences on the structure of a protein [108, 168]. Conditioning predictions on the structure of a known binder would narrow the search space, thereby enhancing the likelihood of identifying successful binders.

Predictive methods for determining which residues in an antibody interact with the antigen, or the paratope, have been shown to improve accuracy by incorporating information on the antibody structure [169, 170]. Knowledge of the paratope has also been used to improve epitope binning [171].

In the next two sections, I give a brief summary of antibody-specific structure prediction methods.

1.5.2 Homology modelling

The global structure of antibodies is similar and therefore can often be accurately predicted using homology modelling. In fact, homology modelling methods have been able to consistently model the majority of the antibody structure with sub-angstrom accuracy [53]. The exception to this is CDR-H3, which due its high sequence and structural diversity, still remains a challenge. See Figure 1.6 for an example of a typical homology modelling pipeline.

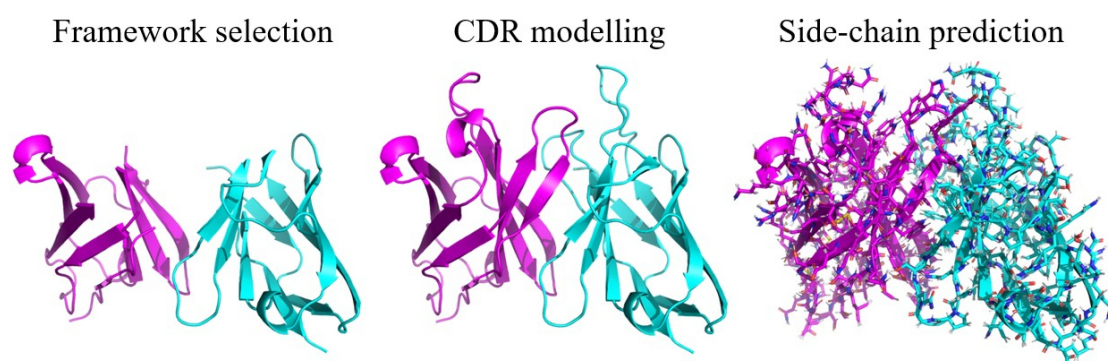


Figure 1.6: A typical homology modelling antibody Fv modelling pipeline. First, the structure of the framework is modelled by selecting a VH and VL framework and a relative orientation between them from already solved structures. Second, the CDR loops are grafted onto the framework. Finally, side chains atoms are modelled by either using rotamer libraries or energy minimisation.

Throughout the work described in this thesis, I use the original version of ABodyBuilder [118] either as a template to model CDR loops (Chapter 2), or as a method to compare against (Chapter 3). ABodyBuilder follows a similar pipeline to the one presented in Figure 1.6. Initially, it selects a set of structures with the highest sequence identity from the Structural Antibody Database (SAbDab) [154] to model the VH and VL frameworks. Then, it identifies a highly identical Fv to replicate the relative VH-VL orientation [55].

CDR loops are modelled onto the framework using FREAD [172], in the order L2-H2-L1-H1-L3-H3. FREAD is a homology modelling tool that searches databases of loop structures for entries predicted to share similar conformations. This is done through the combination of two strategies. First, it ranks them using Environment-Specific Substitution Tables. These tables evaluate the likelihood of two amino acids being interchanged without changing the loop's geometry. Identified loop candidates are then selected based on how effectively the loop anchors graft onto the framework. Any loops without a suitable template are modelled by "ab initio" methods such as Modeller [173], hybrid methods like Sphinx [174], or are classified as unmodelable. Target residues that match those of the template retain their side chain conformations. Meanwhile, side chains without an available conformation from the template are modelled using PEARS [175]. PEARS is a Gaussian mixture model, built from structures in SAbDab to assign the most probable rotamer conformations to each rotatable side chain bond at each IMGT residue position.

1.5.3 Deep learning

Deep learning based approaches have also been shown to improve structure prediction in antibodies. The first method to successfully use deep learning to improve antibody structure prediction was DeepH3 [176], where they used a CNN to predict inter-residue distances and orientations. The predicted inter-residue geometries are then fed into Rosetta [177] to predict the conformation of CDR-H3. This approach was then extended for whole Fv structure prediction with DeepAb [178]. DeepAb was shown to outperform all available antibody structure prediction methods at the time.

The first antibody structure prediction method based on equivariant deep learning was ABlooper [1]. This method is described in depth in Chapter 2. ABlooper was developed before the AlphaFold2 paper and code were released. Since then, there has been a number of equivariant graph neural networks developed for predicting antibody structures. IgFold [179] combines a language model pretrained on antibody sequences [68] with parts of the AlphaFold2 structure module and is shown to outperform AlphaFold-Multimer [146] while being much faster. EquiFold [152] modifies the AlphaFold2 structure module to consider interactions between all rigid groups in the side chain individually and claims to outperform IgFold. Other novel methods for predicting antibody structures include xTrimoABFold [180], tFold-Ab [181], and AbFold [182], all of which claim to be faster or more accurate than the ones before. In Chapter 3 I present our own improved antibody structure prediction method ABodyBuilder2 [2].

The combination of accurate general protein structure prediction [3] and faster yet equally accurate methods to generate antibody structures [2] suggests that we may soon be able to generate accurate antibody-antigen complexes for large sets of antibodies. The current bottleneck is the accuracy of methods for antibody-antigen docking [183]. Our capacity to model antibody-antigen complexes could be improved by incorporating information on predicted paratopes and epitopes into docking methods [184–188], or potentially by end-to-end antibody-antigen complex prediction [146]. Accurate models for these complexes could be used for a broader range of downstream tasks.

Key among these tasks is virtual screening, which could significantly benefit from methods to quickly rank docked antibody-antigen structures. In this context, techniques such as DLAB could play a substantial role [189]. DLAB is a deep learning method trained to re-rank docked antibody-antigen poses. Furthermore, if the availability of antibody-antigen structural data were to increase significantly, it may become feasible to train computational models capable of accurately predicting binding affinity [190, 191]. Accurate methods for antibody binding affinity prediction

would be a significant stride towards fully computational antibody design, potentially transforming the landscape of drug discovery and therapeutic development [36].

1.6 Chapter Walkthrough

Chapter 2 introduces ABlooper, a deep learning model designed to predict the position of backbone atoms for CDR loops in antibodies. ABlooper has the capability to model CDR loops with accuracy comparable to other state of the art methods, and can generate them at unprecedented speeds. However, ABlooper has its limitations, it does not model side-chain atoms and occasionally produces unphysical structures. The rectification of these anomalies requires computationally intensive energy minimisation methods.

Chapter 3 presents ImmuneBuilder, a suite of deep learning models designed for predicting the structures of antibodies, nanobodies, and TCRs. Unlike ABlooper, ImmuneBuilder accurately determines the positions of both backbone and side-chain atoms. Additionally, it predicts the structure of the entire variable domain, not just the CDR loops. I show that ImmuneBuilder’s performance in predicting antibody structures is on par with AlphaFold-Multimer, while being over a hundred times faster.

Chapter 4 introduces KA-Search, a method designed for executing quick and exhaustive searches across billions of antibody variable domains. Searches are based on their amino acid sequence identity across the variable domain, CDRs, or any user-defined antibody region. I illustrate how KA-Search can efficiently search the roughly 2.4 billion antibody sequences in the OAS database in under 30 minutes, and a representative subset of 10 million sequences in less than 9 seconds.

Chapter 5 presents PLAbDab, a database of antibodies derived from patents and literature. PLAbDab houses over 150,000 antibody sequences and 3D models, including 65,000 unique ones, each with a link to its source material. I discuss how the data was gathered and processed, and show how users can search this data using keywords, sequences, or structures.

Finally, in Chapter 6, I summarise the main findings from our work and suggest potential avenues for extension or improvement.

2

ABlooper

Contents

2.1	Chapter abstract	30
2.2	Background	31
2.3	Methods	32
2.3.1	Data	32
2.3.2	Deep learning	33
2.3.3	Loop relaxation	35
2.3.4	Benchmarked methods	36
2.4	Results	36
2.4.1	Using ABlooper to predict CDR loops on modelled anti- body structures	36
2.4.2	Prediction diversity as a measure of prediction quality .	37
2.5	Discussion	39

2.1 Chapter abstract

This chapter is based on the paper “ABlooper: fast accurate antibody CDR loop structure prediction with accuracy estimation” [1]. It introduces ABlooper, an equivariant approach for CDR structure prediction. E(n)-Equivariant Graph Neural Networks (E(n)-EGNNs) are used to build ABlooper due to their speed and simplicity [105]. ABlooper directly predicts the structure of CDR loops onto the antibody framework and can estimate a confidence measure for each predicted loop

by simultaneously predicting multiple structures for each loop and comparing them amongst themselves. ABlooper was developed before details on the AlphaFold2 model [3] were made available, and was the first antibody structure prediction method based on equivariant deep learning.

Since the development of ABlooper, several other equivariant neural networks have emerged as promising approaches for predicting antibody structures. Notable examples include IgFold [179], EquiFold [152], xTrimoABFold [180], tFold-Ab [181], and AbFold [182]. Additionally, the developers of AlphaFold2 have introduced AlphaFold-Multimer [146], a specialised model for proteins complexes. I have also since developed an improved antibody structure prediction method (ABodyBuilder2 [2]) which is described in chapter 3. These more recent methods are not discussed in this chapter, but they are where relevant covered in Chapter 3.

2.2 Background

As described in Chapter 1, knowledge of the structure of antibodies is a key part of biotherapeutic development [192]. However, experimental structure determination is time-consuming and expensive so it is not always practical or even possible to use routinely [110]. Computational modelling tools allow researchers to bridge this gap by predicting large numbers of antibody structures to a high level of accuracy [118, 178].

The overall structure of antibodies exhibits a high degree of similarity (see Section 1.2.5), enabling accurate predictions using homology modelling-based methods (e.g., ABodyBuilder [118]). However, the six complementarity determining regions (CDRs) pose challenges in modelling due to their role in conferring structural diversity required for antigen binding. Among these CDRs, the third loop of the heavy chain (CDR-H3) stands out as the most diverse and hence the most challenging to model accurately [53].

Since the release of AlphaFold2 [3], equivariant neural networks have become widespread in the field of protein structure prediction [139, 140]. For a network to be equivariant with respect to a group, it must be able to commute with the

group action. For rotations, this implies that rotating the input before feeding it into the network will yield the same result as rotating the output. In the case of proteins, using a network equivariant to both translations and rotations in 3D space allows us to learn directly from atom coordinates. This differs from previous methods like TrRosetta [131] or the original version of AlphaFold [130], which predicted invariant features such as inter-residue distances and orientations, which were then used to reconstruct the protein. In this chapter, I investigate an equivariant approach to CDR structure prediction, I chose to use $E(n)$ -Equivariant Graph Neural Networks ($E(n)$ -EGNNs) [105] due to their computational efficiency and simplicity (see Section 1.3.4).

Non-equivariant deep learning approaches had previously shown improvements in antibody structure prediction (see Section 1.5.3), exemplified by DeepH3 [176], an antibody-specific adaptation of TrRosetta. Upon its release, DeepAb [178], an improved version of DeepH3, outperformed all existing antibody structure prediction methods available at the time. Similar to TrRosetta and the original AlphaFold, DeepAb and DeepH3 employ deep learning to predict inter-residue geometries, which are subsequently used in a computationally expensive energy minimisation process to generate the final structure.

2.3 Methods

2.3.1 Data

The data used to train, test and validate ABlooper was extracted from SAbDab [154], a database of all antibody structures contained in the PDB [193]. Structures with a resolution better than $3.0\text{\AA}$ and no missing backbone atoms within any of the CDRs were selected. The resolution cutoff for the training set was chosen to match that used in DeepH3 [176]. The CDRs were defined using the Chothia numbering scheme [39].

For easy comparison with different pipelines, I used the 49 antibodies from the Rosetta Antibody Benchmark as our test set [176, 178]. For validation, 100 structures were selected at random. I ensured that there were no structures with the same CDR sequences in the training, testing and validation sets. Sequence

redundancy was allowed within the training set to expose the network to the existence of antibodies with identical sequence but different structural conformations. This resulted in a total of 3438 training structures.

Additionally, I use a secondary test set composed of 114 antibodies (SAbDab Latest Structures) with a resolution of under 2.3 Å and a maximum CDR-H3 loop length of 20 (same filters used as for the Rosseta Antibody Benchmark), which were added to SAbDab after the initial test, train and validation sets were extracted (8th November 2020 – 24th May 2021). The CDR-H3 length cutoff is set to reduce the likelihood of having loops with multiple conformations in the benchmark sets, as longer loops are known to be more flexible. The test sets are however still a representative sample of antibodies in SAbDab, where the average CDR-H3 length is 12.8 (standard deviation of 5.15). A list containing the PDB IDs of all the structures used in the train, test, and validation sets is given in Appendix A. Due the research efforts to combat the global pandemic during these months, it is noteworthy that 22 out of the 114 antibodies within the SAbDab Latest Structures test set are SARS-CoV-2 binders.

ABodyBuilder was used to build models of all the structures. Structural models were generated using the singularity version of ABodyBuilder [118] (fragment database from 8th July 2021) excluding all templates with a 99% or higher sequence identity. ABlooper CDR models for the test sets were obtained by remodelling the CDR loops on ABodyBuilder models.

2.3.2 Deep learning

ABlooper is composed of five slightly modified E(n)-EGNNs, each one with four layers, all trained in parallel. The number of layers and parallel E(n)-EGNNs blocks was chosen as they were found to be the best performing set of parameters that could fit in an NVIDIA Tesla T4 GPU. The model is trained on the position of the C_{α} -N-C- C_{β} backbone atoms for all six CDR loops plus two anchor residues at either end. E(n)-EGNNs require a starting geometry, so a non-descriptive input geometry is generated by evenly spacing each CDR loop residue on a straight line between its

anchor residues (Figure 2.1). The model is given four different types of features per node resulting in a 41-dimensional vector. These include a one-hot encoded vector describing the amino acid type, the atom type and which loop the residue belongs to. Additionally, sinusoidal positional embeddings are given to each residue to make the model aware of how many amino acids away it is from the anchors. This is necessary as GNNs are by design equivariant to permutations to the order of their nodes. An outline of how E(n)-EGNNs are used within ABlooper is shown in Figure 2.1.

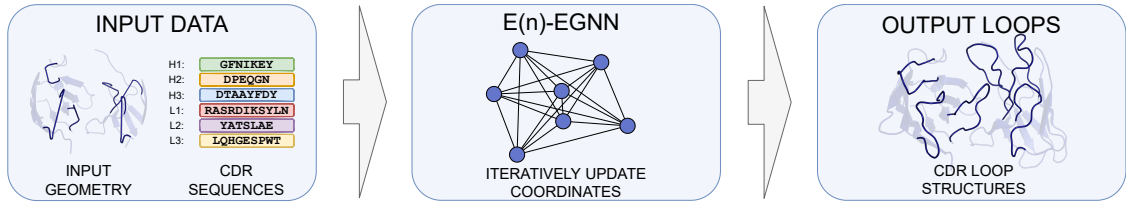


Figure 2.1: Flowchart showing how E(n)-EGNN are used to predict CDR loops in ABlooper. The input geometry for each CDR loop is generated by aligning its residues between their anchors, while the node features are extracted from the loop sequence. Atom coordinates are then iteratively updated using a four layer E(n)-EGNN resulting in a predicted set of conformations for each CDR.

The main difference between our implementation of E(n)-EGNN and the original is the normalisation of relative coordinates before each coordinate update. I found this to enhance stability during training. Additionally, our model does not incorporate edge features. With these modifications, the equations describing the E(n)-EGNN model become:

$$\begin{aligned}
 \mathbf{r}_{ij} &= \mathbf{x}_i^l - \mathbf{x}_j^l \\
 \mathbf{m}_{ij} &= \phi_e(\mathbf{h}_i^l, \mathbf{h}_j^l, \|\mathbf{r}_{ij}\|^2) \\
 \mathbf{x}_i^{l+1} &= \mathbf{x}_i^l + C \sum_{j \neq i} \hat{\mathbf{r}}_{ij} \phi_x(\mathbf{m}_{ij}) \\
 \mathbf{m}_i &= \sum_j \mathbf{m}_{ij} \\
 \mathbf{h}_i^{l+1} &= \mathbf{h}_i^l + \phi_h(\mathbf{h}_i^l, \mathbf{m}_i)
 \end{aligned}$$

Where $\mathbf{x}_i^l$ are the coordinates of node i after layer l and $\mathbf{h}_i^l$ is the 41-dimensional feature vector of node i . In this notation, ϕ denotes a Multi-Layer Perceptron (MLP), C represents a learnable parameter at each layer, and $\hat{\mathbf{r}}_{ij}$ is the normalised distance between points i and j . The edge MLP (ϕ_e) consists of Linear-SiLU-Linear-SiLU layers, yielding 32 features for each pair of nodes. The node and coordinate update MLPs (ϕ_x and ϕ_h) are both composed of Linear-SiLU-Linear layers.

Each of the five E(n)-EGNNs were trained to make predictions independently by minimising the RMSD between their prediction and the crystal structure. The output from the five networks is then averaged to obtain a final prediction. To encourage the final combined prediction of all E(n)-EGNNs to be physically plausible, the L1-loss was used on the final averaged structure. This was composed of five terms between the following pairs of atoms: $C_\alpha^i-C_\alpha^{i+1}$, $C_\alpha^i-C_\beta^i$, $C_\alpha^i-N^i$, $C_\alpha^i-C^i$, C^i-N^{i+1} .

The model was trained in two phases. First it was trained until convergence without the L1-loss term using the RAdam [194] optimiser with a learning rate of 10^{-3} and a weight decay of 10^{-3} . In the second stage, the L1-loss term was added with a weighting of 1.0. For this stage, the model was trained using the Adam [88] optimiser with a learning rate of 10^{-4} and early stopping.

2.3.3 Loop relaxation

During training, ABlooper is encouraged to predict physically plausible CDR loops via the intra-residue atom distance L1-loss term. However, ABlooper occasionally produces loops with incorrect backbone geometries. To enforce correct backbone geometries I relax the predicted loops using a restrained energy minimisation procedure. As the energy function, I use the AMBER14 [195] protein force field with an additional harmonic potential term keeping the positions of backbone atoms close to their original predicted positions. The spring constant of the harmonic potential is set to 10 kcal/molÅ². Energy minimisation is done using the Langevin Integrator in the OpenMM python package [196]. This relaxation step typically results in a small loss in accuracy, but removes clashes and nonphysical bonds from predicted loops.

Since publication, the relaxation step in ABlooper has been updated to include changes introduced in Chapter 3 that reduce issues such as predicting D-stereoisomers or cis-peptide bonds. A more detailed description of these changes is given in Section 3.3.4.

2.3.4 Benchmarked methods

DeepAb structural models were generated using the open source version of the code (available at <https://github.com/RosettaCommons/DeepAb>). As suggested in their paper [178], I generated five decoys per structure. This took around 10 minutes per antibody on an 8-core Intel i7-10700 CPU.

Antibody structures were generated using the open source version of AlphaFold2 (available at <https://github.com/deepmind/alphafold>). I used the "full_dbs" preset and allowed it to use templates from before the 14th May 2020. As AlphaFold2 is intended to predict single chains [3], I predicted and aligned the heavy and light chain independently before comparing to other methods. On a 20-core Intel 6230 CPU this took around 3 hours per antibody modelled.

2.4 Results

2.4.1 Using ABlooper to predict CDR loops on modelled antibody structures

I used ABlooper to predict the CDRs on ABodyBuilder models of the Rosetta Antibody Benchmark (RAB) and the SAbDab Latest Structures (SLS) sets. The RMSD between the C_{α} -N-C- C_{β} atoms in the backbone of the crystal structure and the predicted CDRs for both test sets is shown in Table 2.1.

ABlooper achieves lower mean RMSDs than ABodyBuilder for most CDRs (Table 2.1). By far the largest improvement is for the CDR-H3 loop, where due to the large structural diversity, homology modelling performs worst [118]. ABlooper predicts loops of a similar accuracy to AlphaFold2 and DeepAb for all CDRs except CDR-H3, where ABlooper and DeepAb outperform AlphaFold2. An example of below and above average predictions for each method are shown in Figure 2.2.

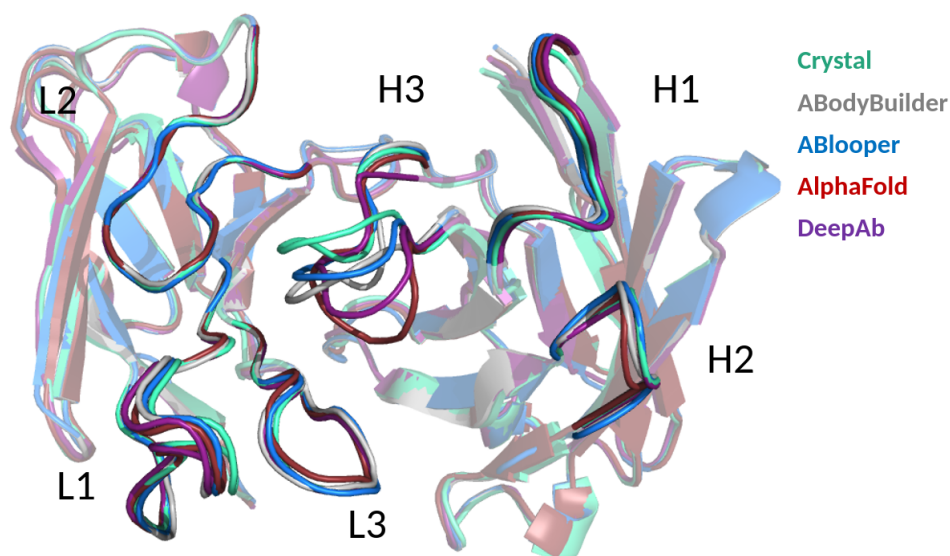
Method	CDR-H1	CDR-H2	CDR-H3	CDR-L1	CDR-L2	CDR-L3
Rosetta Antibody Benchmark						
AlphaFold2*	0.84	0.99	2.87	0.53	0.49	0.95
ABodyBuilder	1.08	0.99	2.77	0.69	0.50	1.12
DeepAb	0.83	0.93	2.44	0.50	0.44	0.85
ABlooper	0.92	1.01	2.49	0.62	0.52	0.97
ABlooper Unrelaxed	0.90	1.03	2.45	0.61	0.51	0.93
SAbDab Latest Structures						
ABodyBuilder	1.24	1.07	3.25	0.88	0.57	1.03
DeepAb*	1.00	0.82	2.49	0.59	0.45	0.90
ABlooper	1.14	0.97	2.72	0.74	0.55	1.04
ABlooper Unrelaxed	1.14	0.99	2.66	0.73	0.54	1.01

Table 2.1: Performance comparison between AlphaFold2, ABodyBuilder, DeepAb and ABlooper for both test sets. The mean RMSD to the crystal structure across each test set for the six CDRs is shown. RMSDs for each CDR are calculated after superimposing their corresponding chain to the crystal structure. RMSDs are given in Angstroms (Å). (*) It is likely that AlphaFold2 used at least some of the structures in the benchmark set during training. Similarly, structures in the SAbDab Latest Structures set may have been used for training DeepAb.

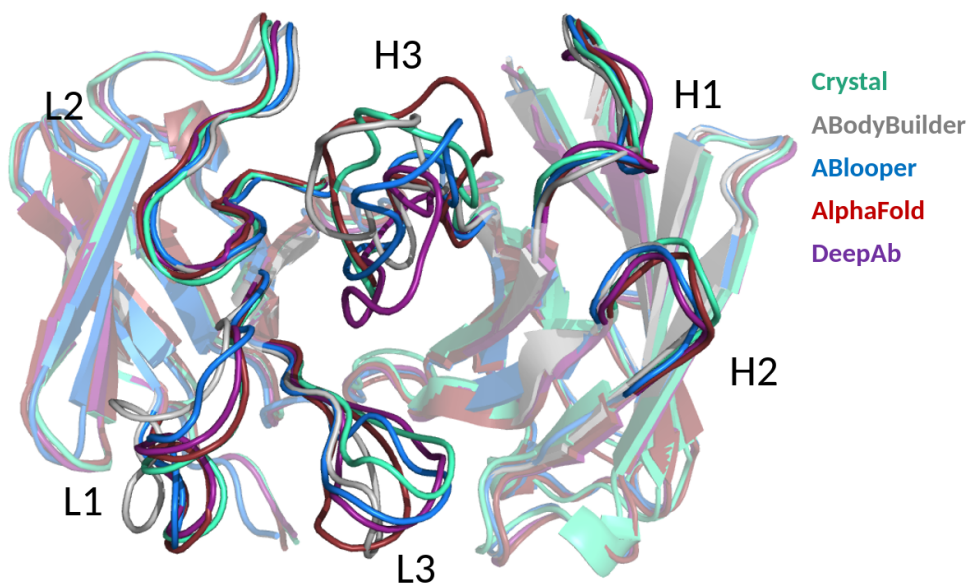
One potential source of error for ABlooper are the model frameworks generated by ABodyBuilder, so in Figure 2.3, I examined its resilience to the small deviations seen in these models and found little to no correlation between framework error and CDR prediction error.

2.4.2 Prediction diversity as a measure of prediction quality

ABlooper predicts five structures for each loop. I found that the average RMSD between predictions can be used as a measure of certainty of the final averaged prediction. If all five models agree on the same conformation, then it is more likely that it will be the correct conformation, if they do not, then the final prediction is likely to be less accurate (Figure 2.4). This allows ABlooper to give a confidence score for each predicted loop. As shown in Figure 2.4D, this score can be used as a filter, removing structures which are expected to be incorrectly modelled by ABlooper. For example, by setting a 1.5Å inter-prediction RMSD cut-off on structures from the Rosetta Antibody Benchmark, the average CDR-H3 RMSD for the set can be reduced from 2.45Å to 2.02Å while keeping 76% of predictions.



(A) Example of above average predictions.



(B) Example of below average predictions.

Figure 2.2: Sample predictions for ABlooper, ABodyBuilder, AlphaFold2 and DeepAb compared against the experimentally determined crystal structures. Each CDR is labelled and the legend showing how the structures produced by each method are coloured is shown on the top right.

Longer CDR loops have been shown to be harder to model [176, 178]. As shown in Figure 2.5, longer loops tend to have higher prediction diversity. This means that accuracy filtering will tend to select shorter loops. However, I find that filtering by prediction diversity is effective even amongst loops of the same length. For

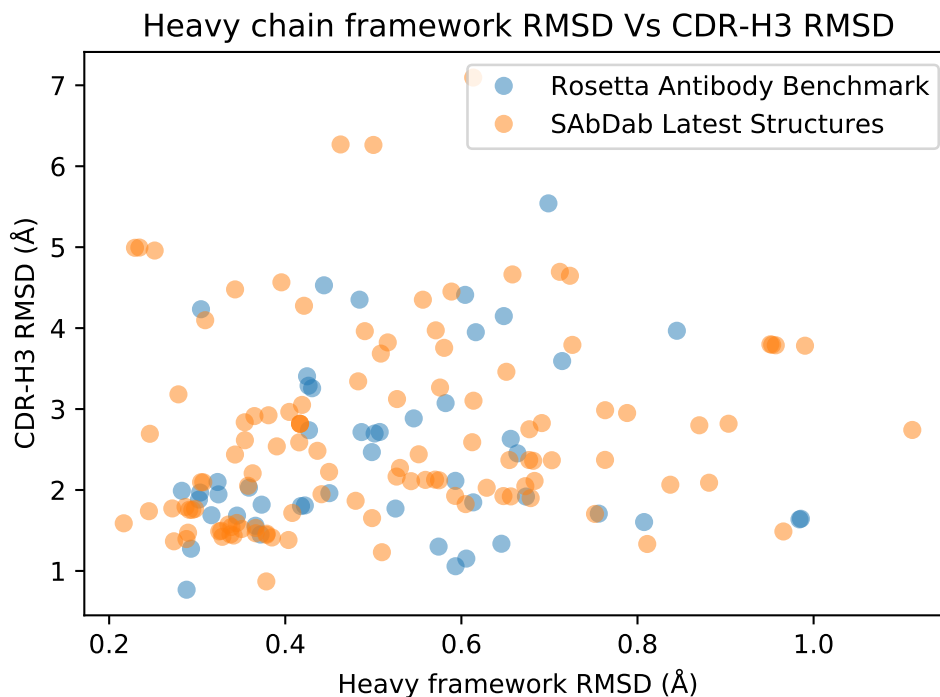


Figure 2.3: Heavy chain framework RMSD of the ABodyBuilder model against CDR-H3 RMSD of ABlooper Predictions. Framework RMSDs were calculated after superimposing the predicted ABodyBuilder framework onto the crystal structure.

example, if we select the 75% best scoring CDR-H3 predictions for each loop length separately, the average RMSD over both test sets drops from 2.60Å to 2.40Å.

2.5 Discussion

I present ABlooper, a fast and accurate tool for predicting the structures of the CDR loops in antibodies. On an NVIDIA Tesla V100 GPU, the unrelaxed version of ABlooper can predict the CDR backbone atoms for one hundred structures in under five seconds. Loop relaxation and side-chain prediction are the most computationally expensive parts of the pipeline taking around ten seconds per structure. ABlooper outperforms ABodyBuilder (a state of the art homology method) and produces antibody models of similar accuracy to both AlphaFold2 and DeepAb, but on a far faster timescale.

By predicting each loop multiple times, ABlooper is capable of producing an accuracy estimate for each generated loop structure. It is not clear whether a

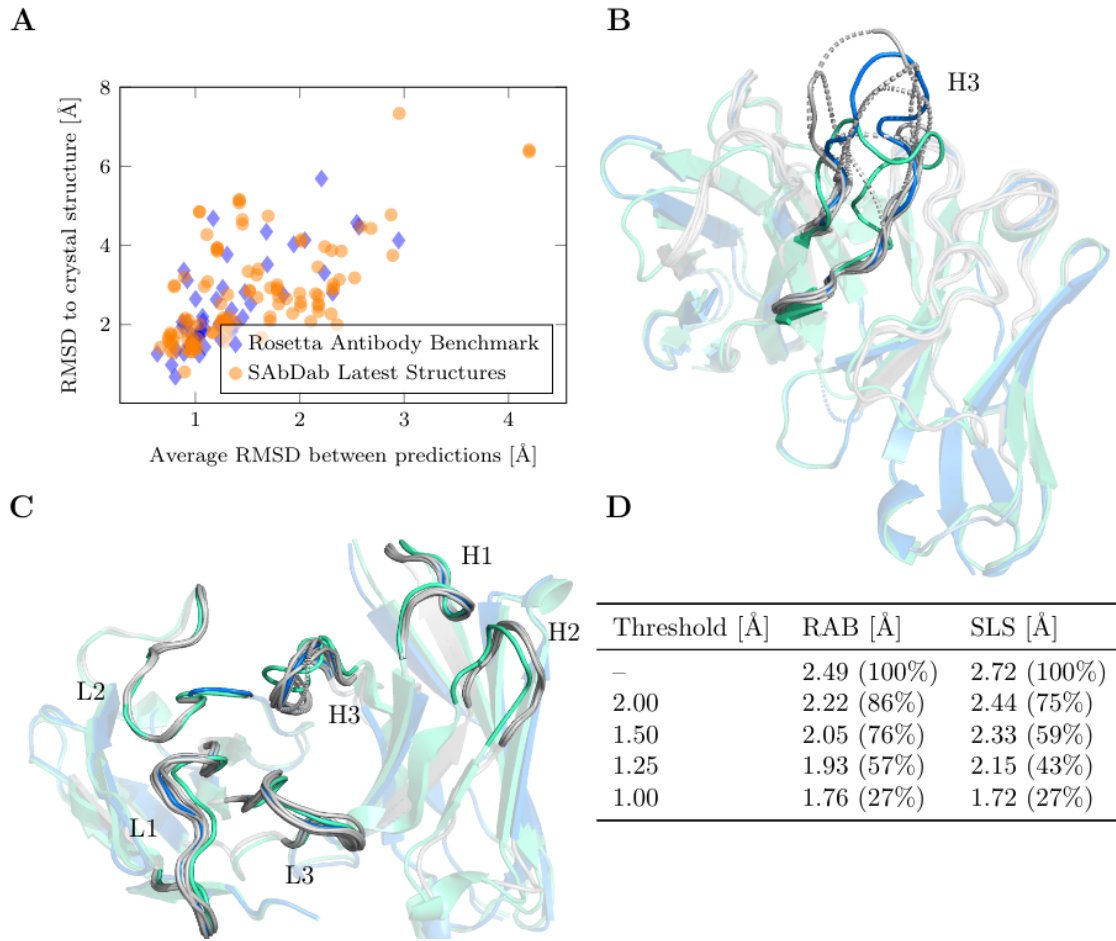


Figure 2.4: (A) CDR-H3 loop RMSD between final averaged prediction and crystal structure compared to average RMSD between the five ABlooper predictions for both the Rosetta Antibody Benchmark and the SAbDab Latest Structures set. (B) An example of a poorly predicted CDR-H3 loop. All five predictions are given in grey, with the final averaged prediction in blue and the crystal structure in green. The predictions from the five networks are very different, indicating an incorrect final prediction. (C) Example of correctly predicted CDR loops. All five predictions are similar, indicating a high confidence prediction. (Colours are the same as in (B)) (D) Effect of removing structures with a high CDR-H3 inter-prediction RMSD on the averaged RMSD for the set. The number of structures remaining after each quality cut-off is shown as a percentage. Data shown for the Rosetta Antibody Benchmark (RAB) and the SAbDab Latest Structures (SLS) sets.

high prediction diversity score is indicative of loops with multiple conformations or underrepresentation of the given loop sequence in SAbDab [154]. However, due to how ABlooper is trained (with the averaged prediction encouraged to be physically plausible), I would expect individual decoys from ABlooper to be unphysical for divergent predictions.

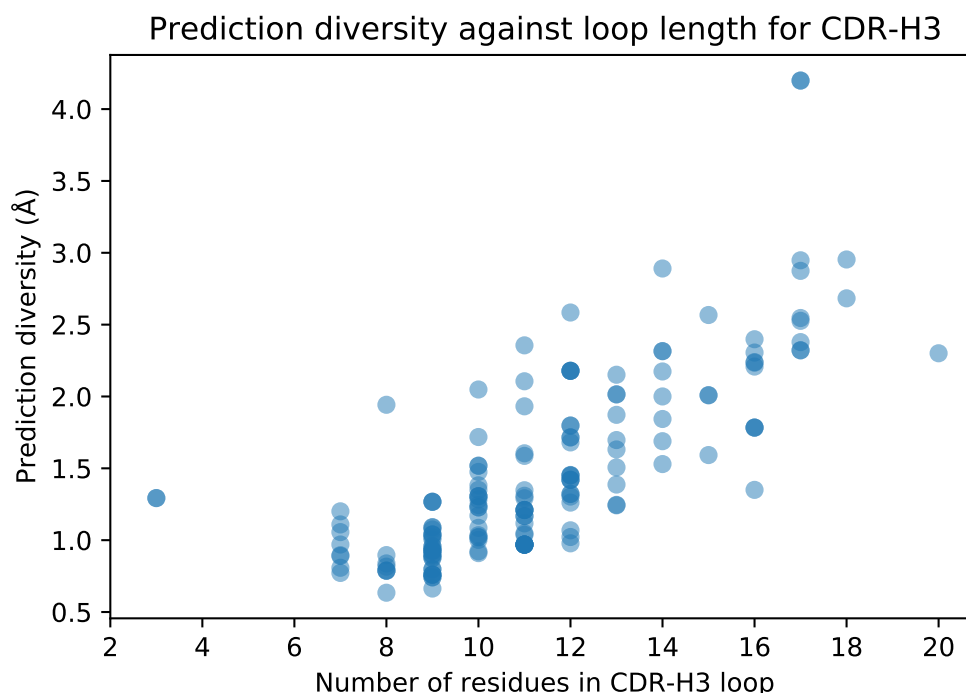


Figure 2.5: Prediction diversity RMSD against the number of amino acids in CDR-H3 loop for all of the structures in the RAB and the SLS datasets.

With the arrival of B-cell receptor repertoire sequencing, the number of publicly available paired antibody sequence data is rapidly increasing [5, 80]. Fast accurate tools such as ABlooper provide the opportunity for structural studies (such as [160]) at previously infeasible scales. The model and inference code for ABlooper is freely available at: <https://github.com/oxpig/ABlooper>.

While ABlooper enabled more accurate modelling of antibody structures at previously unprecedented speeds, it has some limitations. By itself ABlooper does not model side-chain atoms and will sometimes generate unphysical structures, which need to be corrected using computationally expensive methods [197]. The release of AlphaFold2 presented strategies for incorporating side chain prediction into the model and to generate more physically accurate structures. Since then there have been a number of deep learning methods for antibody structure prediction [152, 179–182]. I have also since developed an improved antibody structure prediction method which is described in Chapter 3.

3

ImmuneBuilder

Contents

3.1	Chapter abstract	42
3.2	Background	43
3.3	Methods	46
3.3.1	Data	46
3.3.2	Deep learning architecture	47
3.3.3	Model selection	49
3.3.4	Structural refinement	49
3.3.5	Benchmarked methods	50
3.4	Results	51
3.4.1	Accuracy of prediction	52
3.4.2	Heavy and light chain packing	53
3.4.3	Side chain and chemical surface accuracy	55
3.4.4	Physical plausibility and accurate stereochemistry	56
3.4.5	Computational cost	57
3.4.6	Error estimation	57
3.4.7	Sequence identity against prediction accuracy	58
3.4.8	NanoBodyBuilder2 results	58
3.4.9	TCRBuilder2 results	60
3.5	Discussion	62

3.1 Chapter abstract

This chapter is based on the paper “ImmuneBuilder: Deep-Learning models for predicting the structures of immune proteins” [2]. It describes a set of deep

learning models trained to accurately predict the structure of antibodies (ABodyBuilder2), nanobodies (NanoBodyBuilder2) and T-cell receptors (TCRBuilder2). ImmuneBuilder generates structures with state of the art accuracy while being over a hundred times faster than AlphaFold2. By predicting an ensemble of structures, ImmuneBuilder also gives an error estimate for every residue in its final prediction. The following text is adapted from the paper.

3.2 Background

The adaptive immune system is effective at identifying and neutralising a wide range of pathogens. It achieves this through antigen-specific proteins like antibodies, nanobodies, and T-cell receptors (TCRs), which can bind to an extensive array of pathogens with high specificity (see Section 1.2). The binding site of antibodies and TCRs is concentrated in six loops, three on each of the two Ig domains known collectively as the complementarity determining regions (CDRs). In nanobodies the binding site is concentrated in only three CDR loops on its single Ig domain [83]. These CDR loops display variability in length, composition, and structure, with CDR-H3 being the most variable in antibodies and nanobodies [34] (see Section 1.2.7).

Structures of immune proteins are regularly used during therapeutic design [36, 192] (see Section 1.5.1). However, the time and cost factors involved in experimental structure determination make it impractical to use routinely [110]. Computational modelling has emerged as an effective tool for quickly generating accurate structures for large volumes of sequence data [1, 118, 179] (see Section 1.5). For example, Observed Antibody Space (OAS) contains over two billion antibody heavy chain sequences and is growing rapidly [5, 65]. If this huge amount of sequence data is to even partially be analysed in terms of structure, rapid accurate methods for the prediction of antibody structures are required.

AlphaFold2 is a deep learning method that has revolutionised the field of computational protein structure prediction, achieving near experimental accuracy for a large number of single chain proteins (see Section 1.4.3) [3]. It was later

extended to AlphaFold-Multimer to accurately predict protein complexes [146]. The AlphaFold2 model can be divided into two main steps: In the first step, the Evoformer module is used to extract evolutionary couplings from alignments of many protein sequences into information-rich embeddings. It then uses these embeddings in the structure module to predict the 3D structure of a given protein sequence (see Section 1.4.4 for more details).

Structure prediction methods specific to a certain class of protein tend to outperform more general methods [178, 198]. By using knowledge specific to a type of protein, they can easily predict the conserved regions in that protein allowing greater focus on harder details. For example, DeepH3 was shown to outperform TrRosetta on antibodies [131, 176], while Nanonet obtains results of similar accuracy to AlphaFold2 on nanobodies with a far simpler architecture [199]. More recent examples of this are IgFold [179] and EquiFold [152], where the authors trained antibody-specific models that predict structures of comparable accuracy to AlphaFold-Multimer.

In this chapter I present ImmuneBuilder, a set of deep learning models developed to predict the structure of proteins of the immune system. By training on specific proteins types, we are able to create rapid accurate models, enabling ImmuneBuilder to be routinely used on large sequence data sets. I have built three models: ABodyBuilder2, an antibody specific model, NanoBodyBuilder2, a nanobody specific model and TCRBuilder2 a TCR specific model. I show that these methods perform at least as well as state of the art methods for their respective protein types while predicting structures in a fraction of the time. I also demonstrate that these methods both accurately predict details of the structure and create physically and biologically sensible structures.

The three ImmuneBuilder models are made freely available for download and as web-servers.

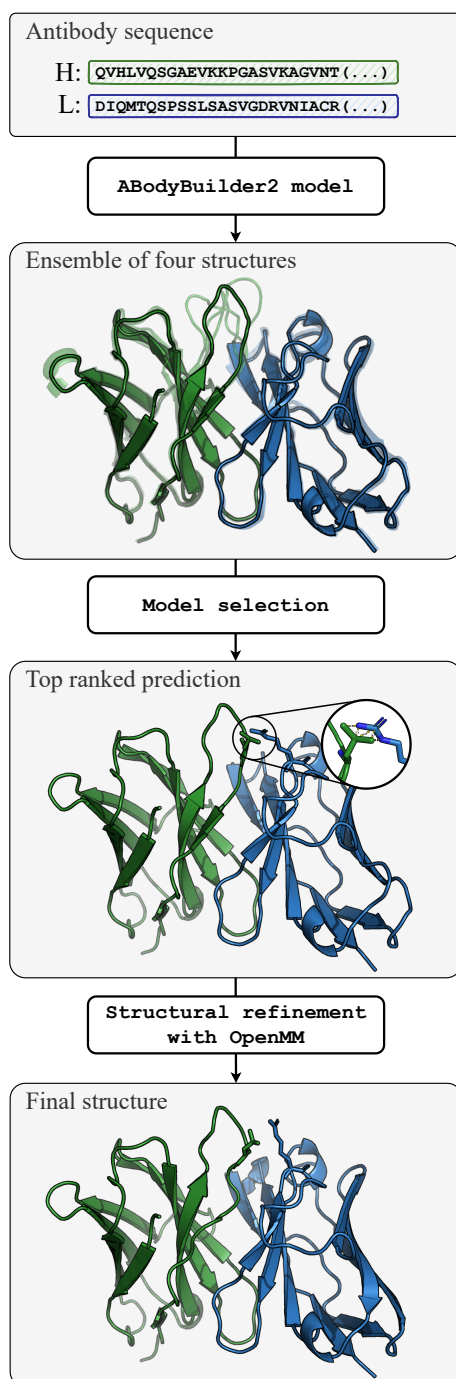


Figure 3.1: Pipeline used to predict structures by ABodyBuilder2. First, the heavy and light chain sequences are fed into four separate deep learning models to predict an ensemble of structures. The closest structure to the average is then selected and refined using OpenMM [196] to remove clashes and other stereo-chemical errors. The same pipeline is used for NanoBodyBuilder2 and TCRBuilder2.

3.3 Methods

In this section I describe in detail the data and models for creating ABodyBuilder2. An overview of the steps used to predict an antibody structure with ABodyBuilder2 is shown in Figure 3.1. The model architecture, training procedure, model selection and structural refinement for NanoBodyBuilder2 and TCRBuilder2 are the same as for ABodyBuilder2.

3.3.1 Data

The data used to train, test, and validate ABodyBuilder2 was extracted from SAbDab [154], a database containing all antibody structures in the PDB [193]. The training data was extracted on the 31st of July 2021 resulting in a total of 7084 structures. Filters were used to ensure structures in the training data had both the VH and VL chains, were not missing residues other than at the start and end of the chain, and had a resolution of 3.5Å or better (this value was chosen as it removes just under 20% of the data). Structures with the same amino acid sequence were kept in the training data to expose the model to antibodies with multiple conformations. As a validation set, I used the set of 49 antibodies in the Rosetta Antibody Benchmark. Structures with the same sequence as antibodies in the validation set were removed from the training set.

For the test data, I extracted all PDB files containing antibody Fv structures in SAbDab added between the 1st of August 2021 and the 1st of June 2022. Only crystal structures resolved by X-ray diffraction and with a resolution better than 2.3Å were kept. A set of non-redundant Fvs were then selected from these and further filtered to remove antibodies with CDR-H3s longer than 22 amino acids and structures with missing residues. These filters are the same that were used to generate the test sets for ABlooper [1] and IgFold [179]. Finally, it was ensured that there were no structures with the same sequence in the test, training, and validation sets. This resulted in the set of 34 Fvs that was used to benchmark ABodyBuilder2 against other methods. It is worth noting, that due to the research efforts to combat the global pandemic, 11 out of these 34 Fvs were from SARS-CoV-2 binding antibodies.

To train NanoBodyBuilder2, 1935 nanobody structures were extracted from SAbDab [156] on the 10th of July 2022, fifty of which were randomly selected as a validation set. To benchmark it against other methods, forty structures released between January and July 2022, with a resolution better than 2.3Å and resolved via X-ray diffraction were selected as a test set. To train TCRBuilder2, the structures of 704 TCR variable domains were extracted from STCRDab [155] on the 20th of May 2022, fifty of which were randomly selected as a validation set. For the benchmark data, the structures of 21 alpha-beta TCRs with a resolution better than 3.5Å and released between January and August 2022 were selected. As in ABodyBuilder2, it was ensured that there were no structures with the same sequence in the train, test and validation sets for NanoBodyBuilder2 and TCRBuilder2.

A full list of PDB codes for structures used in the training, validation and test set for each of the ImmuneBuilder models are given at <https://github.com/oxpig/ImmuneBuilder>.

3.3.2 Deep learning architecture

The architecture of the deep learning model behind ABodyBuilder2 is an antibody-specific version of the structure module in AlphaFold-Multimer with several modifications. Residues are treated as rigid bodies, each one defined by a 3D point in space and a matrix representing its orientation. The input node features are a one-hot encoded representation of the sequence and the input edge features are relative positional encodings. At the start, all residues are set at the origin with the same orientation.

The model is composed of eight update blocks run sequentially. At every iteration, the node features are first updated in a structurally aware way using the Invariant Point Attention layer, and then residue coordinates and orientations are updated using the Backbone Update layer. For further details on how these layers work, see Section 1.4.3 or the original AlphaFold2 paper [3]. Finally, torsion angles for each residue are predicted from node features and are then used to reconstruct an all-atom structure using hard-coded rules. Unlike AlphaFold-Multimer, all

blocks have their own weights. The relevant part of the AlphaFold2 architecture is depicted in Figure 1.5d.

The main term in the loss function is the Frame Aligned Point Error (FAPE) loss (Equation 1.4.3), which quantifies how structurally similar the true and predicted structures are in the local reference frame of each residue. For details see Section 1.4.3. In AlphaFold2, the FAPE loss is clamped at $10\text{\AA}$ focusing on correctly placing residues relative to those closest to it. Similar to AlphaFold-Multimer, a modified version of FAPE loss is used for ABodyBuilder2 in which more focus is given to correctly placing CDR residues relative to the framework. This is achieved by clamping the FAPE loss at $30\text{\AA}$ when it is calculated between framework and CDR residues and at $10\text{\AA}$ otherwise. The final loss term is a sum of the average backbone FAPE loss after every backbone update and the full atom FAPE loss from the final structure. The values of $30\text{\AA}$ and $10\text{\AA}$ were chosen to replicate those used in AlphaFold-Multimer.

As is done in AlphaFold2, a structural violation term is added to the loss function. This penalises nonphysical conformations with a term for bond angles, bond distances, and clashing heavy atoms. In our models, this term was reduced by an order of magnitude with respect to AlphaFold2 as this was found to slightly improve prediction accuracy without significantly harming the physicality of the final prediction. Finally, the side-chain and backbone torsion angle loss from AlphaFold2 is also used.

Each model was trained in two stages. In the first stage, the structural violations term of the loss function was set to zero and a dropout of 10% was used. The RAdam optimiser [194] was used with a cosine annealing scheduler with warm restarts every 50 epochs, learning rates between $1e-3$ and $1e-4$, and a weight decay of $1e-3$. For the second stage, the structural violations loss is added and dropout is set to zero. RAdam is also used for this stage with a fixed learning rate of $1e-4$ and weight decay of $1e-3$. To aid with stability, the norm of gradients is clipped to a value of 0.1 in the second stage of training. For both stages, a batch size of 64 is used and training

is stopped if there was no improvement in the validation set after 100 epochs. On average, training took around four weeks for each model on a single Tesla P100 GPU.

3.3.3 Model selection

ABodyBuilder2 is composed of four deep learning models trained independently to predict antibody structures. To select the best prediction, I align all predicted structures using the Kabsch algorithm [200] and choose as the final prediction the closest one to the average. This reduces the method’s sensitivity to small fluctuations in the training set. It also results in a small improvement in prediction accuracy.

3.3.4 Structural refinement

Although the models are encouraged to predict physically plausible structures during training, they will occasionally produce structures with steric clashes, incorrect peptide bond lengths, or cis-peptide bonds. A restrained energy minimisation procedure with OpenMM is used to resolve these issues. The AMBER14 protein force field [195] with an added harmonic force term to keep the heavy atoms of the backbone close to their original positions is used. In the rare case when two side chain atoms are predicted by the model to be within $0.2\text{\AA}$ of each other, the clashing side chains are deleted and remodelled using `pdifix` [196]. AMBER14 does not explicitly consider chirality, so when the predicted structure contains peptide bonds in the cis configuration, an additional force is added to flip their torsion angles into the trans configuration.

By design, the ABodyBuilder2 deep learning model will always generate amino acids in their L-stereoisomeric form. However, it was found that during energy minimisation residues are occasionally flipped into their D-stereoisomer. To fix this, a method similar to that in [201] is used. First, the chirality at the carbon alpha centre of each D-stereoisomeric residue is fixed by flipping the hydrogen atom. The structure is then relaxed keeping the flipped hydrogen atoms in place before a final minimisation.

3.3.5 Benchmarked methods

I compared ABodyBuilder2 to five other methods: a homology modelling method (the original version of ABodyBuilder [118] (ABB)), one general protein structure prediction method (AlphaFold-Multimer [146] (AFM)), and three antibody-specific methods (ABlooper [1] (ABL), IgFold [179] (IgF) and EquiFold [152] (EqF)). AlphaFold-Multimer was run using the freely available version of the code [146]. It was run using the weights from version 2.2 and without the use of templates. This generated 25 structures per antibody out of which the top ranked was selected for the benchmark. The public version of their respective code bases (as of December 15th 2022) was used to generate EquiFold [152] and IgFold [179] models. As in their paper, Rosetta [177] is used to minimise IgFold models. The original version of ABodyBuilder [118] was run by using the SAB-Box Singularity container (<https://process.innovation.ox.ac.uk/software/p/20120-a/sabbox-singularity-platform---academic-use/1>) from July 2022 excluding all templates with a sequence identity of 99% or higher. ABlooper [1] (version 1.1.2) was run to remodel the CDR loops from the ABodyBuilder predictions. Structures generated by all methods were numbered using ANARCI [37].

I compared TCRBuilder2 and NanoBodyBuilder2 to homology modelling methods (RepertoireBuilder [202] and TCRBuilder [198] for TCR modelling, and MOE [203] and ABodyBuilder [118] for nanobody modelling) and machine learning methods AlphaFold-Multimer [146] for TCRs and AlphaFold2 [3] for nanobodies. The public webserver for TCRBuilder and RepertoireBuilder were used to generate models (<https://opig.stats.ox.ac.uk/webapps/stcrpred/TCRBuilder/> and https://sysimm.org/rep_builder/, respectively). The standard VHH protocol in MOE version 2019.01 was used for nanobody modelling. To ensure a fair comparison, structures with a sequence identity of over 99% were not used as templates for any of the homology modelling methods.

3.4 Results

In this section I will initially focus on the results for ABodyBuilder2 (AB2) on antibodies with the results for NanoBodyBuilder2 (nanobodies) and TCRBuilder2 (TCRs) briefly discussed in sections 3.4.8 and 3.4.9. All three methods show qualitatively similar results.

To compare ABodyBuilder2 to the other methods, I selected a non-redundant set of 34 antibody structures recently added to SAbDAb [154, 156] as a benchmark (see methods). This was done so none of the antibody structures in the benchmark would have been seen during training for any of the benchmarked methods. To give a complete picture of how these methods perform, I carryout a comprehensive benchmark using five different measures. Figure 3.2 shows an example of a prediction by ABodyBuilder2, highlighting important aspects of structural modelling.

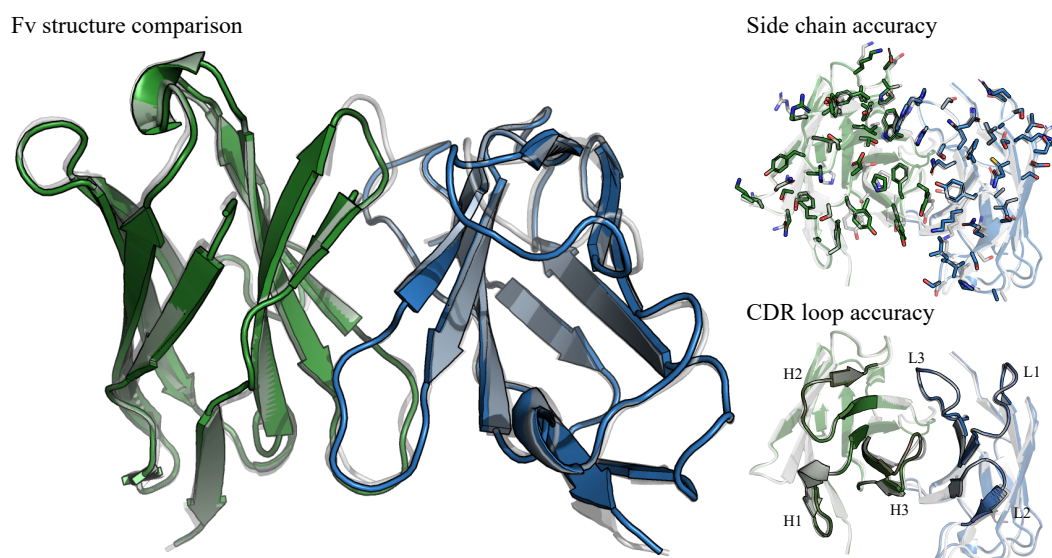


Figure 3.2: Example of an antibody structure predicted with ABodyBuilder2. The heavy chain is shown in green, the light chain in blue and the crystal structure in white. The figure on the left shows the overall Fv structure, demonstrating how ABodyBuilder2 accurately predicts the relative orientation between the heavy and light chain. The figures on the right focus on the structure of CDR loops. The top image showcases the accuracy of ABodyBuilder2 at modelling side chain atoms and the bottom image shows its accuracy at predicting the backbone of CDR loops.

3.4.1 Accuracy of prediction

To measure how accurately the backbone atoms are predicted, the RMSD between predicted and true structures for each antibody region were compared. The RMSD for each CDR and framework is computed by aligning each protein chain to the crystal structure and then calculating the RMSD between the C_α , N, C and C_β atoms. Regions are defined using the IMGT numbering scheme [30]. The results of this analysis are shown in Table 3.1.

The experimental error in protein structures generated via X-ray crystallography has been estimated to be around 0.6Å for regions with organised secondary structure (such as the antibody frameworks) and around 1Å for protein loops [204]. On average, the predicted structures for most of the antibody regions using any method have errors within the range of what would be expected from experimentally resolved crystal structures. The exception to this is CDR-H3, where all methods make the worst predictions.

Method	CDR-H1	CDR-H2	CDR-H3	Fw-H	CDR-L1	CDR-L2	CDR-L3	Fw-L
ABodyBuilder (ABB)	1.53	1.09	3.46	0.65	0.71	0.55	1.18	0.59
ABlooper (ABL)	1.18	0.96	3.34	0.63	0.78	0.63	1.08	0.61
IgFold (IgF)	0.86	0.77	3.28	0.58	0.55	0.43	1.12	0.60
EquiFold (EqF)	0.86	0.80	3.29	0.56	0.47	0.41	0.93	0.54
AlphaFold-M (AFM)	0.86	0.68	2.90	0.55	0.47	0.40	0.83	0.54
ABodyBuilder2 (AB2)	0.85	0.78	2.81	0.54	0.46	0.44	0.87	0.57

Table 3.1: Comparison between ABodyBuilder, ABlooper, IgFold, EquiFold, AlphaFold-Multimer and ABodyBuilder2 at predicting the backbone atoms for each antibody chain. The mean RMSD to the crystal structure across the test set for each of the six CDRs and framework (Fw) is shown. RMSDs are given in Angstroms (Å). The best performing method for each metric is highlighted in bold.

ABodyBuilder2 and AlphaFold-Multimer are the most accurate methods at predicting the structure of CDR-H3 (RMSD of 2.81Å and 2.90Å respectively). EquiFold, IgFold and ABlooper generate structures with CDR-H3 loops around 10% less accurate than ABodyBuilder2 and AlphaFold-Multimer. The worst method for predicting CDR-H3 loops is the original version of ABodyBuilder, showcasing how deep learning has improved our ability to model.

In Figure 3.3, the CDR-H3 RMSD for each antibody in the test set is compared for each pair of benchmarked methods. In the majority of cases, ABodyBuilder2 is consistently better for most antibodies in the benchmark. The exception to this is AlphaFold-Multimer, where there appears to be a number of antibodies for which AlphaFold-Multimer outperforms ABodyBuilder and vice versa.

3.4.2 Heavy and light chain packing

As described in Section 1.2.5, in antibodies the binding site sits across the heavy and light chain variable regions (VH and VL). With half of the CDRs on each chain, the relative VH-VL orientation can have an impact on the structure of the binding site.

To quantify how accurate each method is at predicting the relative orientation between chains, in Table 3.2 I show the average absolute error in the five angles (HL, HC1, HC2, LC1, LC2) and distance (dc) that fully characterise VH-VL orientation [55]. A brief description of how these values are defined is given in Figure 1.2, for a more complete description see [55].

Method	HL	HC1	LC1	HC2	LC2	dc
Xtal	1.18	0.48	0.75	0.62	0.80	0.11
ABB	0.83	1.09	0.82	1.81	0.90	0.10
ABL	0.80	0.97	0.83	1.70	0.90	0.12
IgF	0.63	0.91	0.71	1.40	0.74	0.14
EqF	0.64	0.98	0.64	1.68	0.72	0.11
AFM	0.67	0.74	0.69	1.45	0.63	0.11
AB2	0.64	0.90	0.66	1.37	0.61	0.12

Table 3.2: Comparison of VH-VL orientation between ABodyBuilder (ABB), ABlooper (ABL), IgFold (IgF), EquiFold (EqF), AlphaFold-Multimer (AFM) and ABodyBuilder2 (AB2). Values shown were calculated using ABangle [55]. HL, HC1, LC1, HC2, LC2 and dc are defined in Figure 1.2 and [55]. The error in the angles HL, HC1, LC1, HC2 and LC2 is shown in degrees with the error in the distance dc given in Angstroms (Å). The average standard deviation observed in antibodies experimentally resolved over five times is shown for comparison (Xtal). The best performing method for each metric is highlighted in bold.

As an upper bound for the accuracy of predicted structures, the average standard deviation of the VH-VL orientation measurements in 55 antibodies with structures resolved over five times is shown in Table 3.2. In the original study [55], the vector

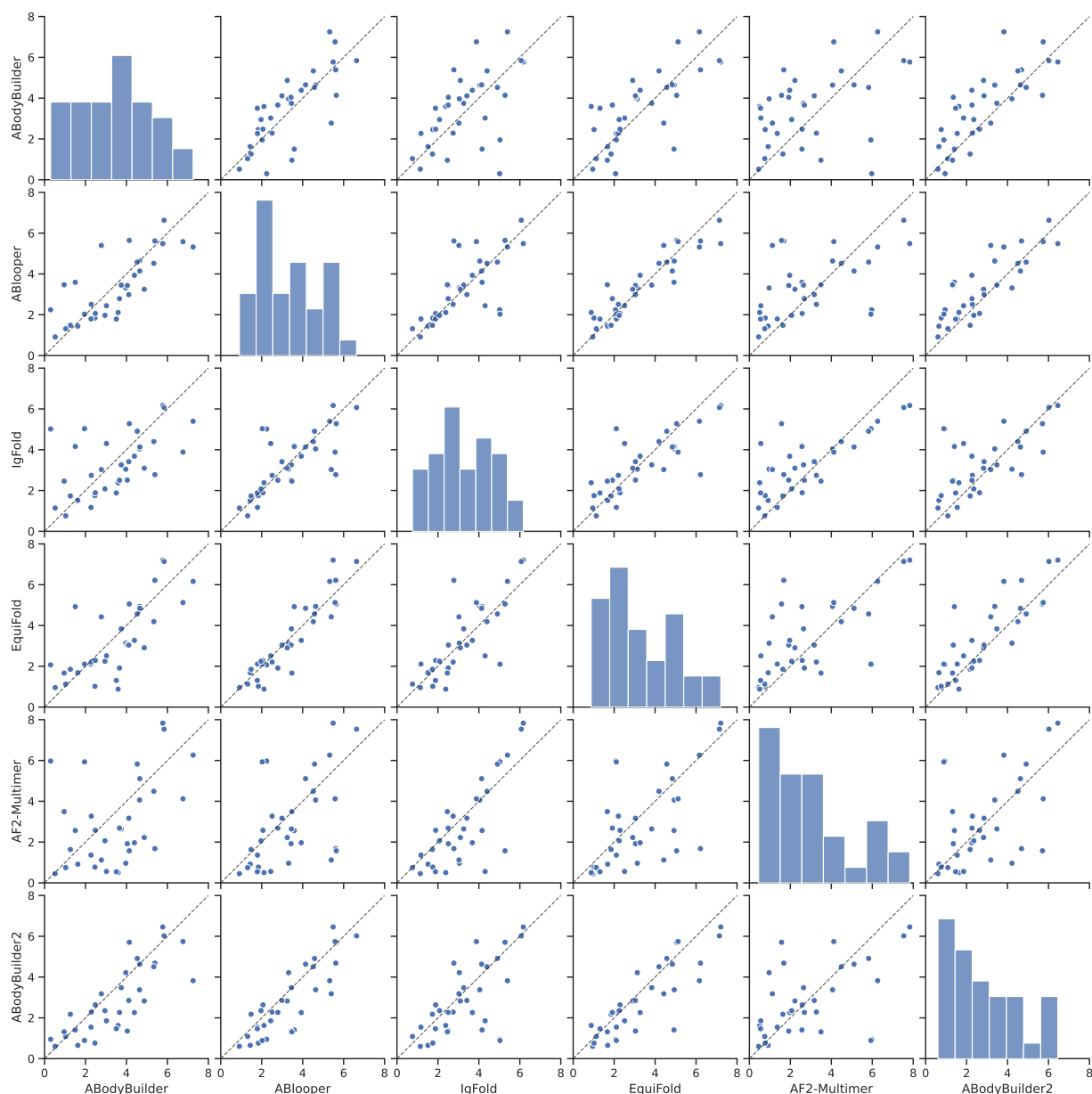


Figure 3.3: Comparison of CDR-H3 RMSDs for each antibody in the test set for each of the benchmarked structure prediction methods. On the diagonal, a histogram showing the distribution of CDR-H3 RMSDs for each method is shown. Off diagonal, scatter plots comparing the ability of each method to model each CDR-H3 in the benchmark is shown. RMSDs are given in Angstroms ($\text{\AA}$)

dc was chosen as an axis as it was found to be the most conserved amongst antibody structures. All of the benchmarked methods predict this distance with very high accuracy. All methods are also accurate at predicting the angles, with small errors with respect to what is observed in experiments. However, small deviations in these angles will still have an impact on the structure of the binding site.

3.4.3 Side chain and chemical surface accuracy

During binding, the antigen will mostly form interactions with side chain atoms on the antibody surface. Therefore to be able to study antigen binding, predicted antibody structures must accurately model the position of side chain atoms and whether they are exposed on the surface or buried. To benchmark the accuracy of side-chain modelling I use a method similar to [175], where a side chain torsion angle is considered correct if it is within 40 degrees of the true conformation. The original implementation of ABodyBuilder will occasionally fail to predict a side chain, this is treated as an incorrect prediction. A residue is labelled as buried if its relative solvent accessibility (calculated as described in [205]) is below 7.5%. The results of this analysis are given in Table 3.3 for ABodyBuilder2.

Method	χ 1	χ 2	χ 3	χ 4	E/B
ABB	0.81	0.77	0.63	0.56	0.92
ABL	0.75	0.70	0.60	0.53	0.90
IgF	0.77	0.66	0.54	0.52	0.91
EqF	0.84	0.71	0.61	0.55	0.94
AFM	0.85	0.77	0.59	0.58	0.91
AB2	0.85	0.78	0.63	0.52	0.91

Table 3.3: Comparison of surface accuracy for ABodyBuilder (ABB), ABlooper (ABL), IgFold (IgF), EquiFold (EqF), AlphaFold-Multimer (AFM) and ABodyBuilder2 (AB2). Values show the accuracy when modelling each of the first four torsion angles of the side chain (χ) if they exist. E/B gives the accuracy at predicting whether a residue is exposed or buried. The best performing method for each metric is highlighted in bold.

As ABlooper and IgFold are deep learning methods that only predict the backbone (leaving side chain prediction to OpenMM [196] and Rosetta [177] respectively), it is perhaps not surprising that they are the least accurate at modelling the chemical surface. EquiFold, AlphaFold-Multimer and ABodyBuilder2, all of which output all-heavy-atom structures, predict the χ 1 and χ 2 side chain atoms with high accuracy while struggling to model longer side chains. The original implementation of ABodyBuilder predicts side chains with comparable accuracy to AlphaFold-Multimer and ABodyBuilder2. All methods are highly accurate at predicting whether a residue is exposed or buried, EquiFold is the most accurate.

3.4.4 Physical plausibility and accurate stereochemistry

Although deep learning models are trained on crystal structures, they will occasionally predict conformations that are very rare or do not occur in nature. I next check for the presence of steric clashes, cis-peptide bonds, D-amino acids, or bonds with nonphysical lengths in the models generated by each method. For bond lengths, only the peptide bond is checked as all other bond lengths are fixed to their literature value in all benchmarked methods but ABlooper and EquiFold.

Method	Peptide bond	D-amino acid	Cis-bond	Clash
Xtal	0	0	0	0
ABB	19	0	3	9
ABL*	1	0	1	1
IgF	0	0	51	10
EqF	271	4	2	765
AFM	0	0	0	0
AB2	0	0	0	0

Table 3.4: Quality check for models generated using ABodyBuilder (ABB), ABlooper (ABL), IgFold (IgF), EquiFold (EqF), AlphaFold-Multimer (AFM) and ABodyBuilder2 (AB2). The errors observed in experimentally resolved crystal structures (Xtal) are also shown for comparison. A peptide bond length is considered to be incorrect if it is more than 0.1Å away from its literature value. Peptide bonds including a proline were not included in the calculation of cis-isomers. Two non-bonded heavy atoms are considered to be clashing when they are closer than 0.63 times their Van-der-Waals radius [175]. (*) Calculated using the latest version of ABlooper, updated since publication to reduce the number of D-amino acids and cis-isomers.

ABodyBuilder2 and AlphaFold-Multimer both generate structures of comparable quality to experimentally resolved ones, whereas IgFold appears to generate a number of cis-peptide bonds and clashes even after being refined with Rosetta [177]. EquiFold does not use an energy based method to refine its predicted structures and hence all of the structures it generates are unphysical. This shows that a refinement step is still necessary to ensure structures generated by deep learning based methods are realistic (Table 3.4).

3.4.5 Computational cost

The original version of AlphaFold-Multimer is by far the most computationally expensive of the benchmarked methods. It requires over one terabyte of sequence data and takes around three hours to generate one structure when run on five CPUs. Large speed-ups can be obtained by reducing the size of the sequence database, using faster sequence alignment algorithms, or using GPUs [150, 151]. Even with these modifications it takes around thirty minutes on a GPU to generate a single structure. All other methods benchmarked can be run on five CPUs in under a minute, with the fastest being EquiFold due to its lack of a refinement step. This makes them all well suited for high throughput structural modelling of next-generation sequencing data. ABodyBuilder2 can also be sped up significantly by using a GPU taking around five seconds to generate an antibody structure on a Tesla P100.

3.4.6 Error estimation

ABodyBuilder2 predicts four structures for each antibody. I found that the diversity between predictions, as in ABlooper, can be used to estimate the uncertainty in the final prediction. If the structures predicted by all four models disagree in a region then the final prediction for this region is likely to be incorrect. This allows ABodyBuilder2 to give a confidence score for each residue that can be used to filter for incorrectly modelled structures. In Figure 3.4 I show how the root mean squared predicted error (RMSPE) for CDR-H3 residues correlates with CDR-H3 RMSD.

A low predicted error does not necessarily indicate an accurate structure. However, a high predicted error works as a good filter for removing inaccurate models. For example, if a predicted error cut-off of around 1Å is set for the current benchmark, it would remove 5 structures with an average RMSD of 4.46Å. The average CDR-H3 RMSD for the remaining set would then be 2.53Å. It is unclear whether a high predicted error is indicative of an antibody with multiple observed conformations, or one whose structure is underrepresented in the training data.

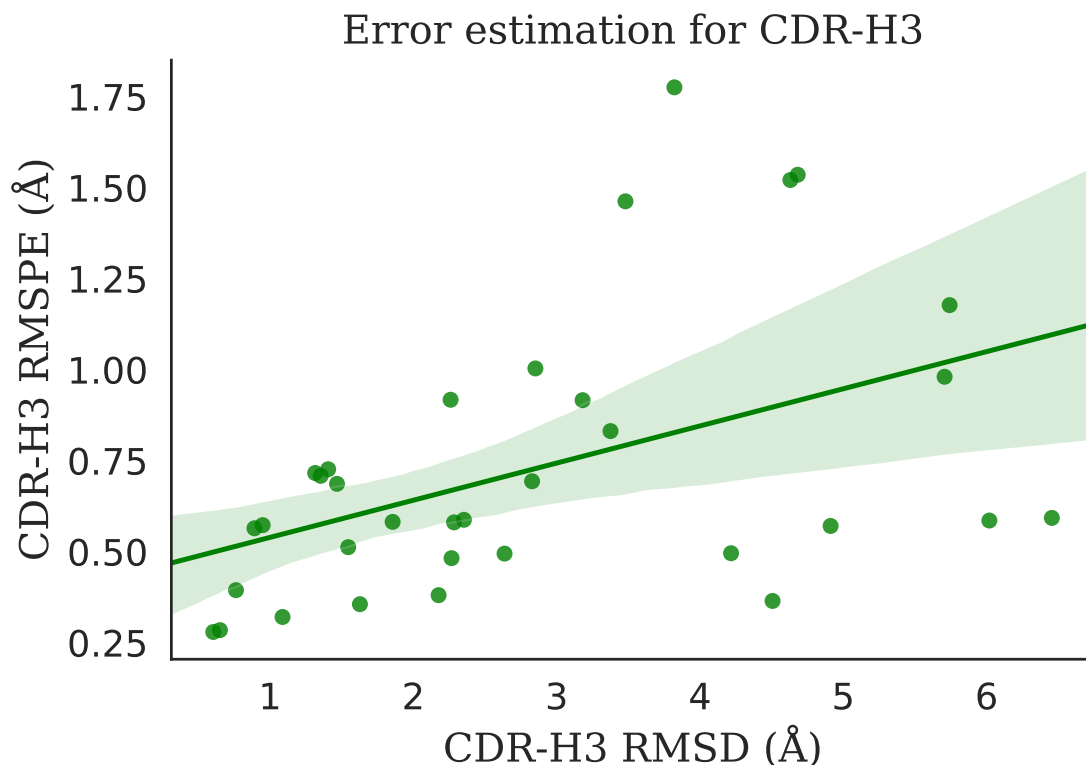


Figure 3.4: Scatter plot showing the CDR-H3 RMSD against the root mean squared predicted error for all structures in the benchmark. The Pearson correlation coefficient between CDR-H3 RMSD and RMSPE is 0.5, indicative of a moderate correlation. The linear fit is included to highlight the existence of a positive correlation. The line shown is the best fit with the 95% confidence interval shown as the shaded area around it.

3.4.7 Sequence identity against prediction accuracy

Antibody structures with an identical heavy and light chain sequence to those in the ABodyBuilder2 training set were excluded from the benchmark set. For each antibody in our benchmark, the most similar antibody in our training set has a sequence identity with values ranging from 62% to 99.5% (the latter having a single insertion in CDR-H3). Figure 3.5 shows that having highly identical sequences in the training set does not necessarily improve the models ability to accurately predict CDR-H3.

3.4.8 NanoBodyBuilder2 results

In this section I compare NanoBodyBuilder2, our nanobody-specific method, to three other tools used for nanobody structure prediction. I compared against two

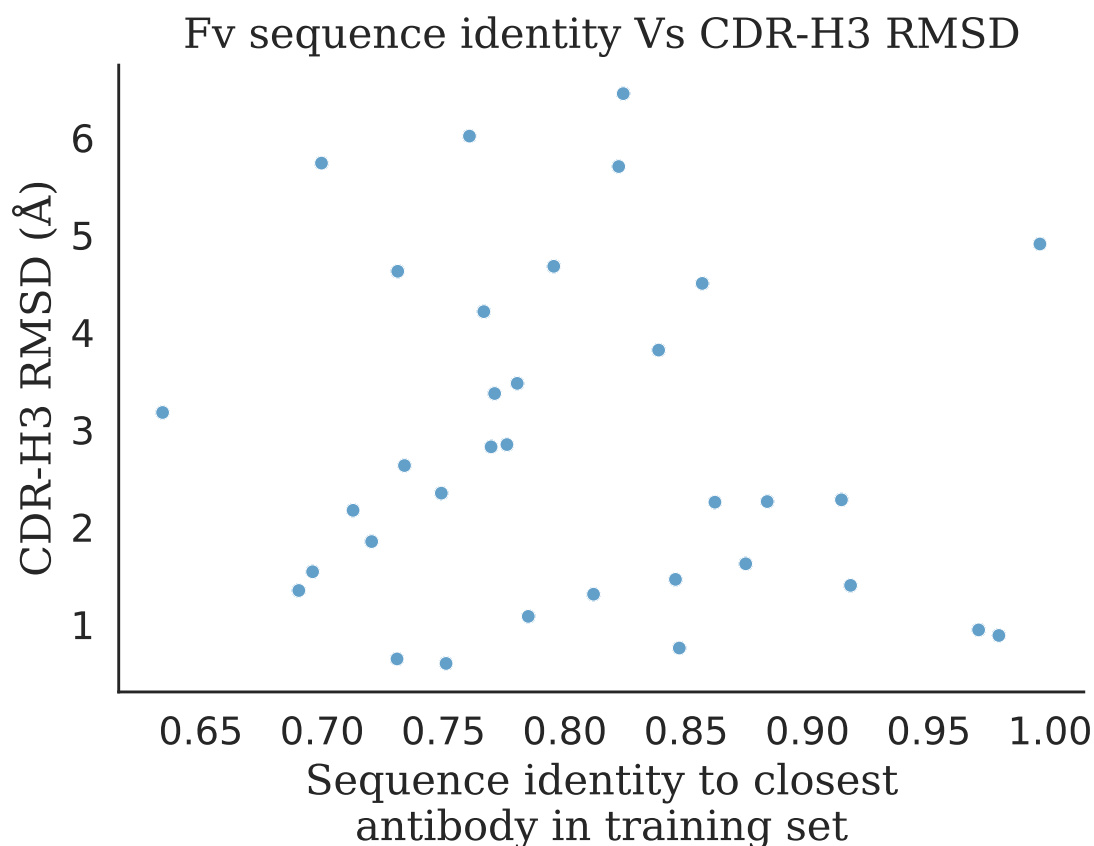


Figure 3.5: CDR-H3 RMSD versus maximum sequence identity to the training set for ABodyBuilder2.

homology modelling methods (the original version of ABodyBuilder [118] and MOE [203]), and one general protein structure prediction method (AlphaFold2 [3]). As a benchmark, I selected a non-redundant set of forty nanobody structures recently added to SAbDAb [156] (see methods).

In Table 3.5, I report results on the same metrics as were used for antibodies. The exceptions to this are metrics involving the light chain, such as heavy and light chain packing angles or light chain regional RMSDs.

NanoBodyBuilder2 predicts the backbone structure of nanobodies with higher accuracy than all other benchmarked methods. The biggest improvement is in the CDR3 loop, which all methods model least accurately. This can also be seen in Figure 3.6 where NanoBodyBuilder2 consistently outperforms other methods at predicting CDR-H3. NanoBodyBuilder2 also predicts the chi (χ) side chain angles with an accuracy comparable to AlphaFold2 and the original version of

Method	CDR1	CDR2	CDR3	Fw
ABodyBuilder	2.96	2.08	5.08	1.09
MOE	2.67	1.99	4.90	1.19
AlphaFold2	2.08	1.35	3.44	0.82
NanoBodyBuilder2	1.98	1.37	2.89	0.79

Method	χ 1	χ 2	χ 3	χ 4	E/B
ABodyBuilder	0.72	0.73	0.54	0.63	0.93
MOE	0.68	0.63	0.43	0.49	0.92
AlphaFold2	0.78	0.74	0.51	0.63	0.94
NanoBodyBuilder2	0.77	0.70	0.53	0.58	0.93

Method	Peptide bond	Clash	D-amino acid	Cis-bond
ABodyBuilder	49	0	8	21
MOE	0	7	6	0
AlphaFold2	0	0	0	0
NanoBodyBuilder2	0	0	0	0

Table 3.5: Comprehensive benchmark between ABodyBuilder, MOE, AlphaFold2 and NanoBodyBuilder2 for predicting nanobody structures. In the first table, the mean RMSD to the crystal structure across the test set for each of the three CDRs and the framework is shown. In the second, I show the accuracy at modelling each of the first four torsion angles of the side chain (χ) and the accuracy at predicting whether a residue is exposed or buried (E/B). The third table shows the total number of the stereochemical errors found in the predicted structures.

ABodyBuilder. In terms of speed, AlphaFold2 takes around one hundred times longer to generate a nanobody structure than any of the other methods.

Figure 3.7 depicts a comparison of the structures of an antibody heavy chain predicted by ABodyBuilder2 and NanoBodyBuilder2. While both models produce similar structures for the framework, the predicted conformation of the CDR-H3 loop differs significantly. ABodyBuilder2 predicts the loop to adopt an erect conformation to avoid collisions with the light chain, while NanoBodyBuilder2 predicts a less compact form.

3.4.9 TCRBuilder2 results

In this section, I compare TCRBuilder2 against three other methods for TCR structure prediction. I benchmarked it against two homology modelling methods (RepertoireBuilder [202] and the original version of TCRBuilder [198]) and a general protein structure prediction method (AlphaFold-Multimer [146]). Additionally, I

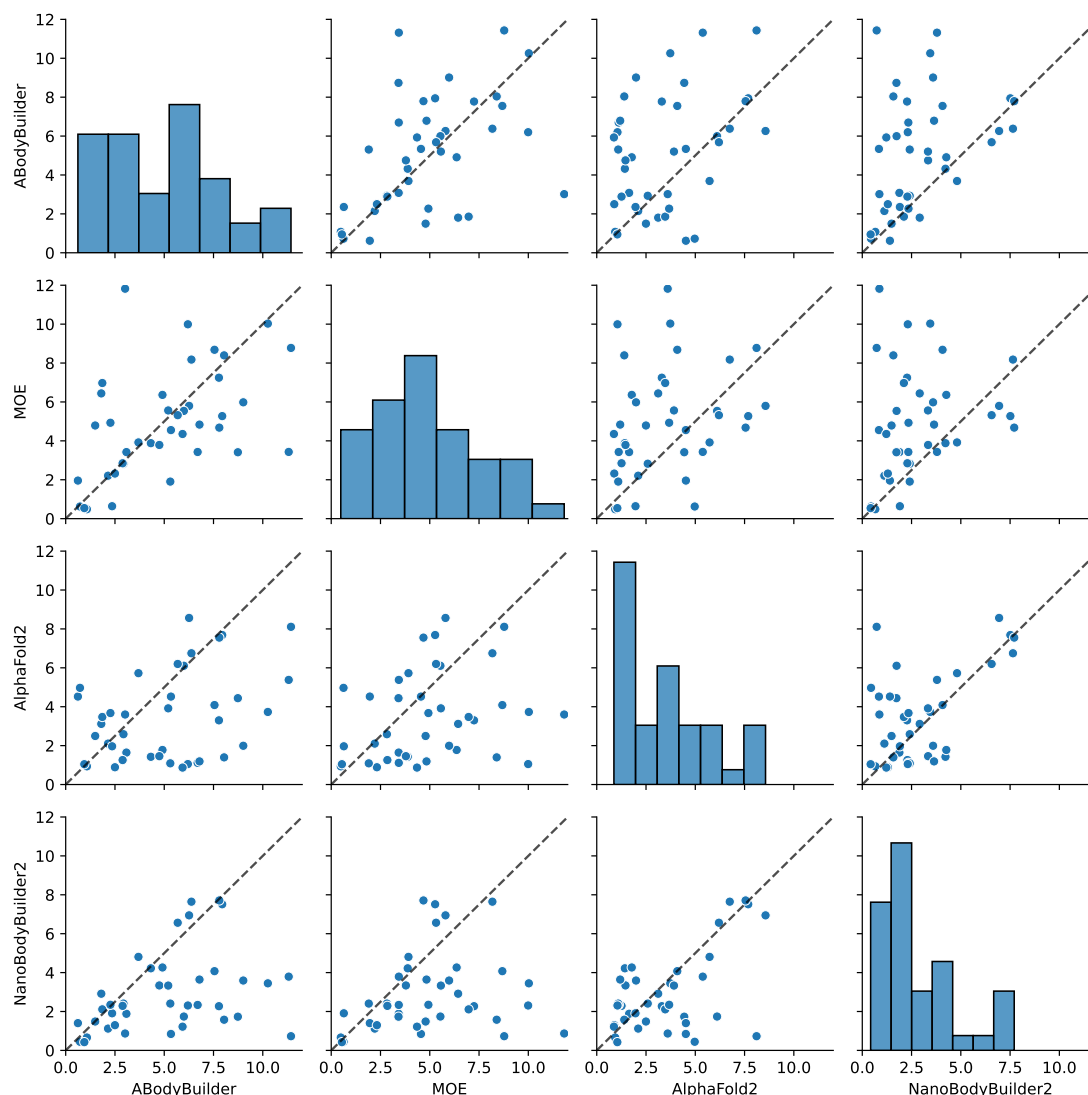


Figure 3.6: Comparison of CDR-H3 RMSDs for each nanobody in the test set for each of the benchmarked structure prediction methods. On the diagonal, a histogram showing the distribution of CDR-H3 RMSDs for each method is shown. Off diagonal, scatter plots comparing the ability of each method to model each CDR-H3 in the benchmark is shown. RMSDs are given in Angstroms ($\text{\AA}$)

compared it to our antibody-specific method (ABodyBuilder2) to showcase the benefits of training only on TCRs. The results for the TCR benchmark on the same metrics as the ones used for antibodies are shown in Table 3.6.

ABodyBuilder2 fails to accurately model TCR structures, with the biggest errors being the modelling of the relative orientation between the alpha and beta chains. Both TCRBuilder2 and AlphaFold-Multimer predict structures with a

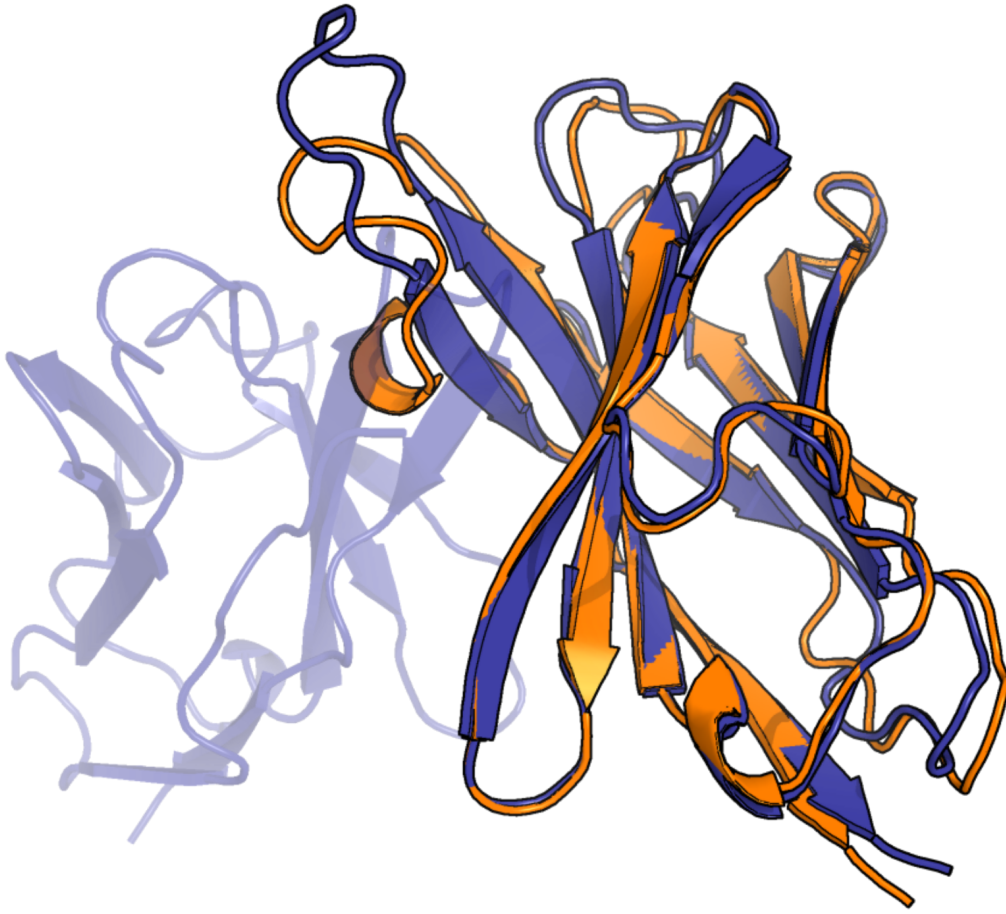


Figure 3.7: Example of an antibody heavy chain as predicted by ABodyBuilder2 and NanoBodyBuilder2. The NanoBodyBuilder2 prediction is shown in orange and the ABodyBuilder2 prediction is shown in blue.

comparable accuracy, outperforming RepertoireBuilder and the original version of TCRBuilder. In terms of run time, TCRBuilder2 takes about as long as ABodyBuilder2, RepertoireBuilder and TCRBuilder, which are all over a hundred times faster than AlphaFold-Multimer.

Figures 3.8 and 3.9 show pairwise comparisons of the accuracy of modelling CDR- α 3 and CDR- β 3 between the benchmarked methods.

3.5 Discussion

I present ImmuneBuilder, a set of three open source and freely available tools for modelling immune proteins capable of rapidly generating accurate antibody, TCR,

Method	CDR- α 1	CDR- α 2	CDR- α 3	Fw- α	CDR- β 1	CDR- β 2	CDR- β 3	Fw- β
TCRBuilder	1.60	1.31	2.89	0.87	0.99	0.90	3.12	0.81
RepertoireBuilder	1.35	1.00	2.64	0.75	0.86	1.59	2.77	1.05
AlphaFold-M	1.25	0.96	1.84	0.69	0.75	0.65	1.94	0.82
ABodyBuilder2	3.49	6.57	3.14	2.89	3.27	3.77	3.48	3.65
TCRBuilder2	1.34	0.93	1.85	0.90	0.74	0.63	1.93	0.67

Method	HL	HC1	LC1	HC2	LC2	dc	Peptide bond	Clash
TCRBuilder	4.80	3.07	2.17	4.03	1.77	0.32	74	29
RepertoireBuilder	8.34	3.09	2.60	3.66	2.79	0.45	115	587
AlphaFold-M	2.95	1.45	1.97	2.83	1.51	0.36	0	0
ABodyBuilder2	74.26	25.65	20.71	63.19	27.42	1.91	0	0
TCRBuilder2	3.32	1.88	2.80	2.44	1.94	0.28	0	0

Method	χ 1	χ 2	χ 3	χ 4	E/B	D-amino acid	Cis-bond
TCRBuilder	0.70	0.69	0.53	0.53	0.90	0	6
RepertoireBuilder	0.73	0.66	0.49	0.54	0.92	0	11
AlphaFold-M	0.77	0.69	0.50	0.52	0.92	0	0
ABodyBuilder2	0.58	0.52	0.36	0.39	0.83	0	0
TCRBuilder2	0.76	0.67	0.50	0.47	0.91	0	0

Table 3.6: Comprehensive benchmark between ABodyBuilder2, TCRBuilder, AlphaFold-Multimer and TCRBuilder2 for predicting TCR structures. The top table shows the mean RMSD to the crystal structure across the test set for each of the six CDRs and frameworks. The CDRs and framework (Fw) regions are labelled α for the alpha chain and β for the beta chain. The second table shows the mean absolute error in each of the six ABangles [55] and the number of unphysical peptide bonds and stereochemical clashes found in predicted structures. In the third table, the accuracy at modelling each of the first four torsion angles of the side chain (χ), the accuracy at predicting whether a residue is exposed or buried (E/B), and the number of D-amino acids and cis-isomers found in predicted structures are shown. For a more in depth description of each metric, see Sections 3.4.1-3.4.4 .

and nanobody structures. ImmuneBuilder can produce structures of antibodies and TCRs with accuracy comparable to AlphaFold-Multimer while being over a hundred times faster and without the need for large sequence databases or multiple sequence alignments. ABodyBuilder2 is shown to be the most accurate of the antibody-specific tools and the only one to consistently predict structures with correct stereochemistry.

The comparison with homology modelling methods, such as ABodyBuilder, shows the benefits that deep learning has brought to the field of antibody structure predictions. However, all methods still struggle to accurately predict the conformation of CDR-H3, suggesting that models capable of predicting multiple conformations may be required to accurately capture this loop. Deep learning methods also still

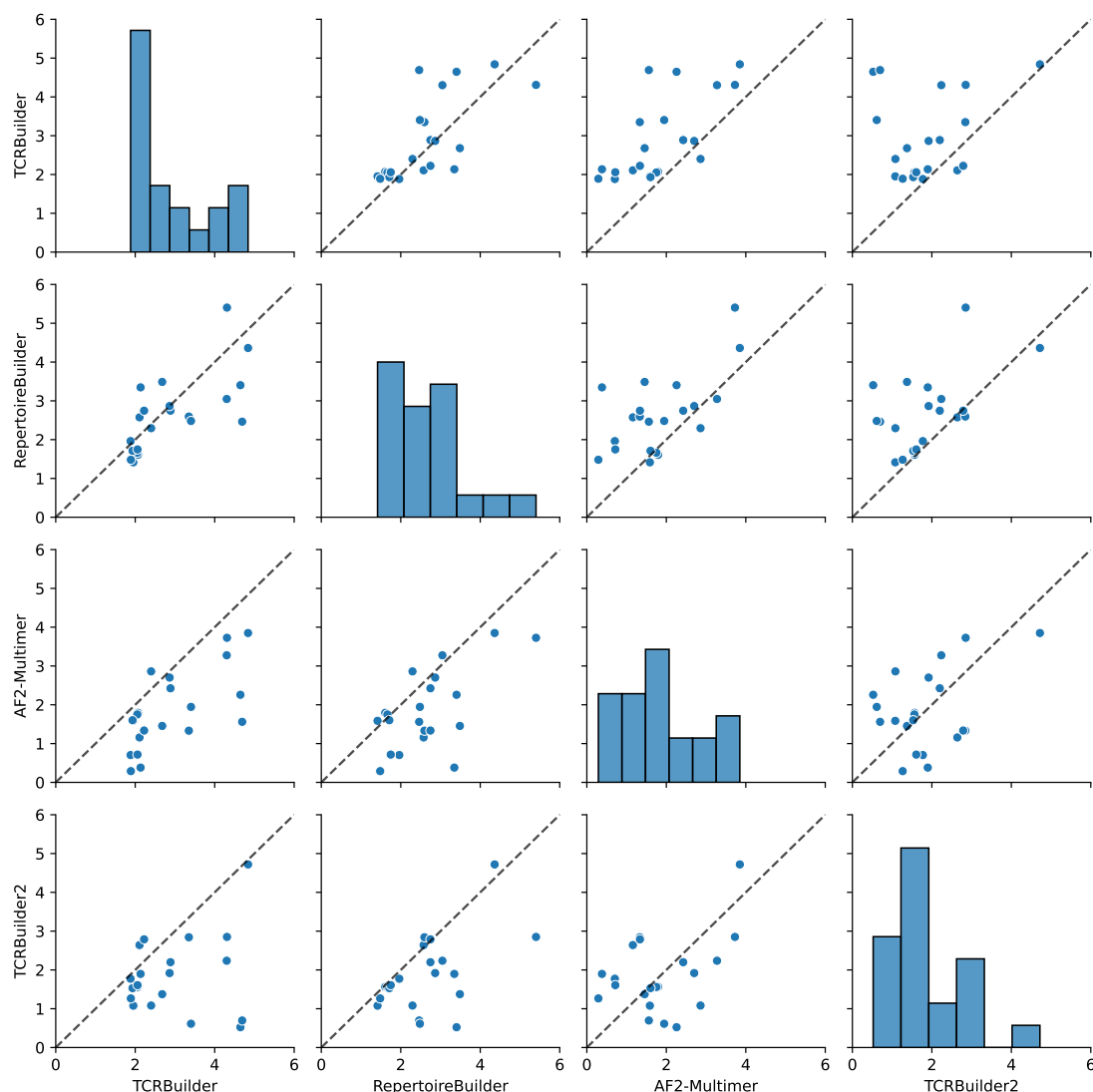


Figure 3.8: Comparison of CDR- α 3 RMSDs for each TCR in the test set for each of the benchmarked structure prediction methods. On the diagonal, a histogram showing the distribution of CDR- α 3 RMSDs for each method is shown. Off diagonal, scatter plots comparing the ability of each method to model each CDR- α 3 in the benchmark is shown. RMSDs are given in Angstroms ($\text{\AA}$)

struggle to consistently predict physically plausible structures. This challenge can be addressed by using physics-based methods, such as restrained energy minimisation, but for fast methods like ABodyBuilder2 this greatly increases computational cost.

By measuring the variability between predictions, the ImmuneBuilder models are able to provide an error estimate for each residue. In combination with its prediction speed and accuracy, the ability to filter for incorrect models make it a

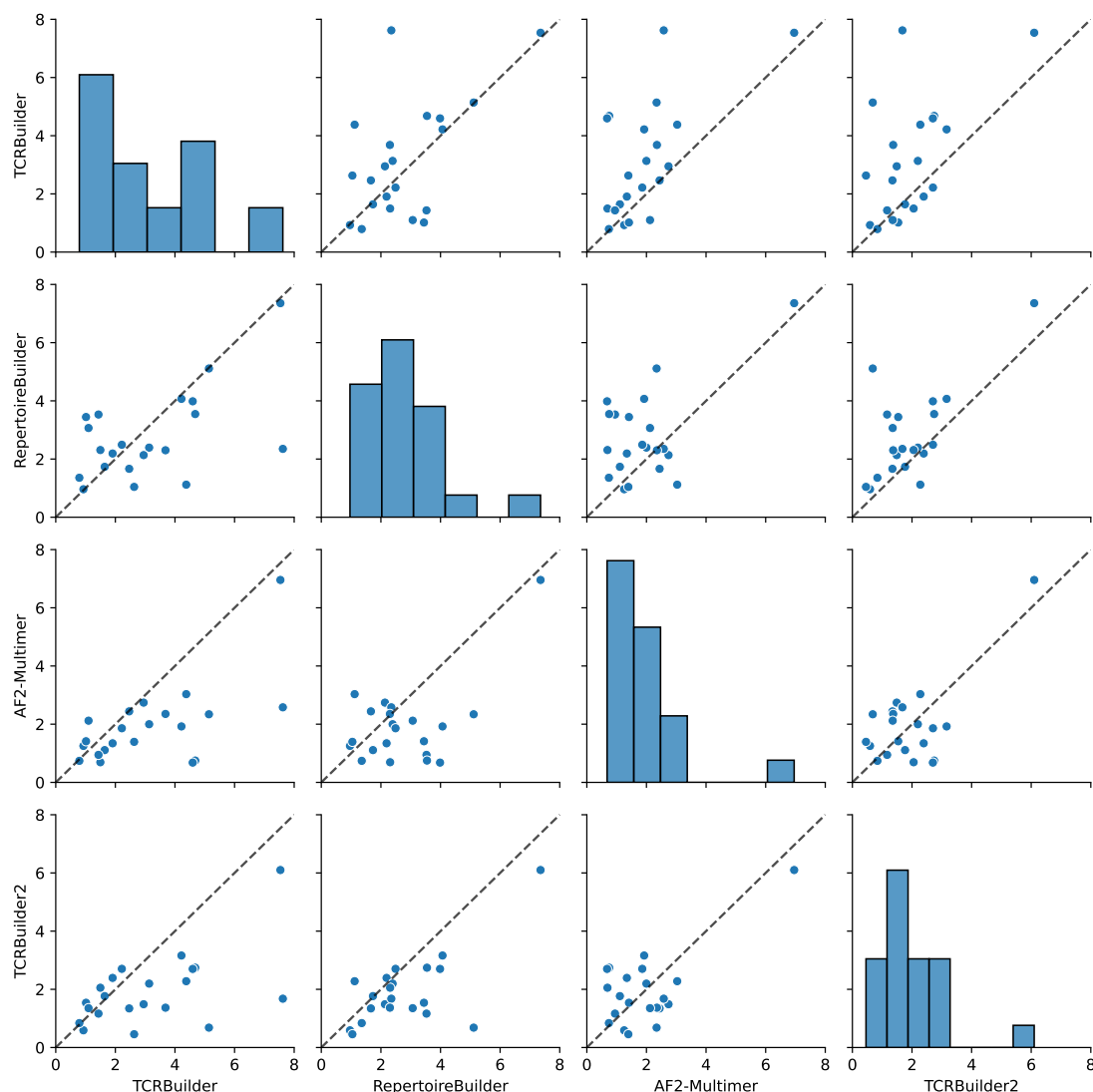


Figure 3.9: Comparison of CDR- β 3 RMSDs for each TCR in the test set for each of the benchmarked structure prediction methods. On the diagonal, a histogram showing the distribution of CDR- β 3 RMSDs for each method is shown. Off diagonal, scatter plots comparing the ability of each method to model each CDR- β 3 in the benchmark is shown. RMSDs are given in Angstroms ($\text{\AA}$)

useful tool for incorporating structural information into data from next-generation sequencing experiments.

To further demonstrate the usefulness of ImmuneBuilder, I predicted the structure of around 148 thousand non-redundant paired antibody sequences from OAS [5] and make these freely available at (<https://doi.org/10.5281/zenodo.7258553>). These models could be used to incorporate structural information in studies of the

human antibody repertoire [75], or to identify human antibodies with a similar structure to a binder of interest [160, 161].

The speed of ImmuneBuilder allows for accurate structural studies at far larger scales than what was possible before. However, for very large scale databases, such as the 2 billion single chain sequences in OAS [65], structural modelling times are still prohibitive. With the rapidly growing adoption and improvements to next-generation sequencing methods, tools to rapidly search the data they generate will be essential for using this data in drug discovery. In Chapter 4 I describe KA-Search, a tool for rapidly and exhaustively searching repertoire data for antibodies with the most identical sequence.

4

KA-Search

Contents

4.1	Chapter abstract	67
4.2	Background	68
4.3	Method	73
4.3.1	Data preprocessing	73
4.3.2	Identity calculation	75
4.3.3	Sensitivity and speed comparison	76
4.4	Results	77
4.4.1	Computational speed of KA-Search	77
4.4.2	Comparison with other common sequence identity search tools	78
4.4.3	Immune repertoire mining with the COVOX-253 antibody	81
4.5	Discussion	83

4.1 Chapter abstract

This chapter is based on the paper "KA-Search: Rapid and exhaustive sequence identity search of known antibodies" [4]. I am a co-first author on the paper alongside Tobias Olsen. My contributions to KA-search were the pre-alignment step and optimisations to the sequence identity calculation. The analysis described in the results section was mostly done by Tobias Olsen, but has been included in this thesis to demonstrate use cases for the KA-Search algorithm. The following

text is adapted from the paper.

As described in Chapter 1, antibodies with similar amino acid sequences, especially across their complementarity-determining regions, often share properties [192]. Finding that an antibody of interest has a similar sequence to naturally expressed antibodies in healthy or diseased repertoires is a powerful approach for the prediction of antibody properties, such as immunogenicity or antigen specificity. However, as the number of available antibody sequences is now in the billions and continuing to grow, repertoire mining for similar sequences has become increasingly computationally expensive [5, 65]. Existing approaches are limited by either being low-throughput, non-exhaustive, not antibody specific, or only searching against entire chain sequences. Therefore, there is a need for a specialised tool, optimised for a rapid and exhaustive search of any antibody region against all known antibodies, to better utilise the full breadth of available repertoire sequences.

In this chapter I present Known Antibody Search (KA-Search) [4], a tool that allows for the rapid search of billions of antibody variable domains by amino acid sequence identity across either the variable domain, the complementarity-determining regions, or a user defined antibody region. We show KA-Search in operation on the ~ 2.4 billion antibody sequences available in the OAS database. KA-Search can be used to find the most similar sequences from OAS within 30 minutes and a representative subset of 10 million sequences in less than 9 seconds. We give examples of how KA-Search can be used to obtain new insights about an antibody of interest.

4.2 Background

For the development of therapeutic antibodies, much work is done to find mutations that improve antibody binding, such as mutations in the paratope that increase the binding affinity to a desired epitope [206]. Also important are mutations that do not change binding, but instead improve other properties of potential therapeutic antibodies. This includes improving their developability profile by removing suboptimal properties, such as undesirable post-translational modification

sites [207] or potential antibody aggregation [208] (see Section 1.2.2). For non-human derived antibodies, issues with immunogenicity are also common, requiring mutations to make the antibody more human-like and thereby reduce the immunogenicity [24]. However, the huge space of antibody sequences and possible mutations makes it a complicated challenge to find the correct mutations to make in order to achieve the desired binding specificity and properties.

A technique which has shown promise for exploring the mutational space of antibodies is immune repertoire mining [73, 209–211]. In immune repertoire mining, an antibody of interest is compared against natural antibody repertoires to find identical or highly similar antibodies. This is useful for finding mutations which could improve binding affinity, as antibodies with few differences in the paratope, derived from patients with the disease of interest, might yield better binders. Finding antibodies in nature with an identical paratope is a powerful method for getting insight into which mutations could improve an antibody’s developability profile or reduce its immunogenicity, without changing its binding properties [73, 160, 209–211] (see Figure 4.1A).

Similarity between antibodies can be measured in different ways. The most common ones are via sequence identity or structural similarity [212, 213]. With a protein’s function being preserved in the structure, structural similarity is often superior for finding proteins with analogous functions, such as antibodies binding the same epitope (see Chapter 5) [160, 214]. However, with orders of magnitude more sequence data available than structural data, a sequence identity search enables the exploration of a much larger space. Available sequence data is also more diverse, as next generation sequencing of B-cell receptors (BCR) is routinely being applied to study adaptive immunity, generating sequences from a range of species [215–217] and from individuals with differing disease states [218, 219]. Furthermore, continuous improvements in high-throughput sequencing methods and increased adoption by research labs means that the amount and diversity of sequence data is rapidly increasing [10, 62] (see Section 1.2.6).

The Observed Antibody Space (OAS) is a database that aims to clean and collate data from next generation sequencing studies into a single source. As of January 2023, OAS contains sequences of the V domain for ~ 2.4 billion unpaired heavy and light antibody chains. These sequences are derived mostly from humans and mice, but also include sequences from rabbits, rats, rhesus', camels and humanised mice. While the size of OAS is promising from a scientific perspective, its scale and continuous growth, visualised in Figure 4.1B, make effectively mining it a challenge. Though sequence identity calculations are simple, without software specially optimised for the task, the computational cost of exhaustively searching OAS or any other large antibody sequence databases is becoming prohibitive. There is therefore a need for specialised tools to search this space now and in the future.

There exist many tools for searching large datasets of protein sequences for similar sequences, for example BLASTp [221], CD-HIT-2D [222], and newer methods such as MMseqs2 [223]. However, these tools are all designed around searching a diverse set of proteins and not specifically antibody sequences. BLASTp finds similar sequences by searching for high-scoring 3-mers for a query within a set of target sequences. This scoring is done using a substitution matrix, such as BLOSUM62 [224]. For target sequences with exact matched 3-mers, the alignment is then extended in both ends until the score decreases and ranked based on their expect value. BLASTp is much faster than performing a pairwise alignment with the Smith-Waterman algorithm between the query and each target sequence; however, BLASTp does not guarantee optimal alignments [221]. To further increase speed, both CD-HIT-2D and MMseqs2 use fast prefiltering steps to remove target sequences with low identity to the query, thereby reducing the number of pairwise alignments to make, a computational expensive step. CD-HIT-2D prefiltering removes target sequences that have an estimated similarity to the query below a specified threshold. Simplified, the estimation is based on two sequences of certain lengths requiring to share a minimum number of k-mers of different sizes, in order to be above a specific sequence identity. After prefiltering, pairwise alignment is performed on the remaining sequences [225]. Prefiltering with MMSeqs2 is also based on comparing

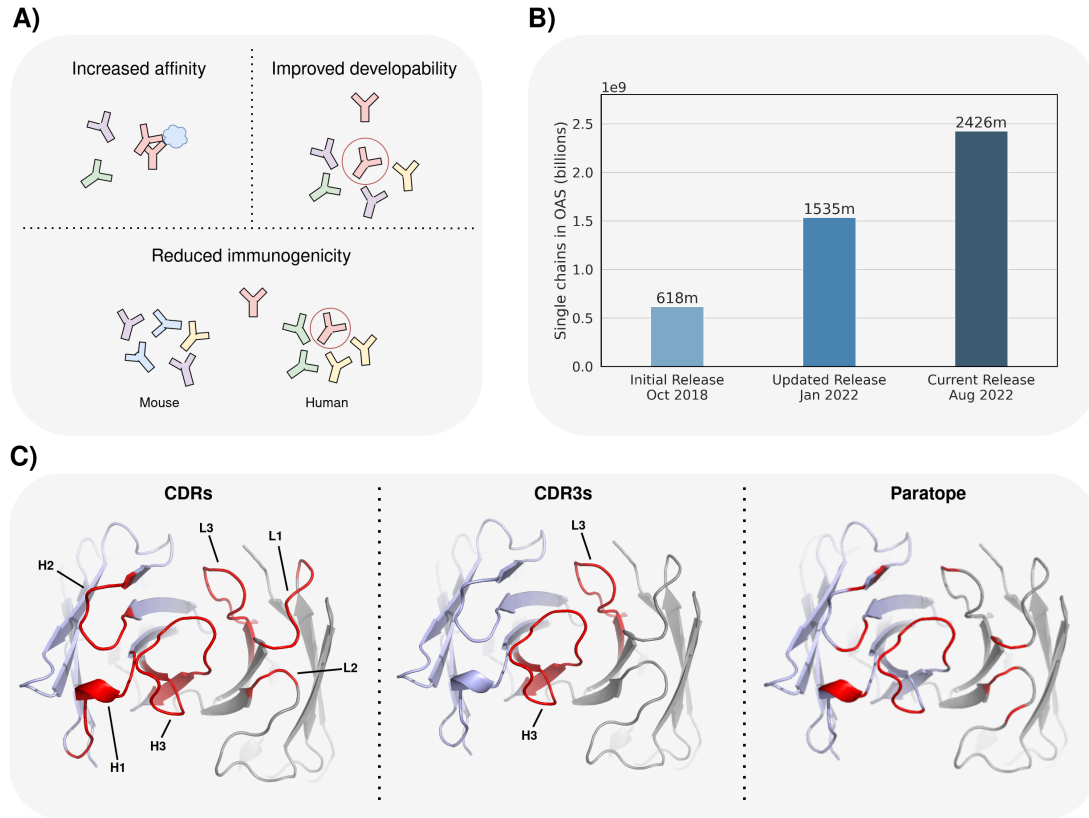


Figure 4.1: **A)** To correctly identify relevant mutations for optimising a given antibody for increased affinity, an improved developability profile or reduced immunogenicity, a huge space of possible mutations needs to be searched. As similar antibodies often bind to the same epitope but with different strengths, immune repertoire mining can be used to find similar antibodies with potentially better binding affinity. Further, immune repertoire mining can be used to find antibodies with the same binding but with mutations improving their developability profile or reducing their immunogenicity. **B)** Overview of the available number of single chain antibody variable domain sequences in the Observed Antibody Space database over time. **C)** Highlight (red) of different specific search regions, useful for finding similar binders. The antibody variable domain is derived from PDB structure 7JOO and the CDRs are annotated using IMGT numbering [220]. Heavy chain complementarity-determining region (CDR)1, 2 and 3 are denoted as H1, H2 and H3, and light chain CDRs as L1, L2 and L3. The paratope was defined as any residue within 4.5Å of the antigen (in this case an inducible T-cell costimulator) and consists of IMGT position 35, 36, 57, 58, 64, 66 and 109-113 on the heavy chain and IMGT position 37, 38, 55, 56, 66 and 114 on the light chain.

k-mers between the query and target sequence. However, instead of exact matches like CD-HIT-2D, MMseqs2 uses a BLAST-like approach of matching k-mers with a BLOSUM62 score above a certain threshold. For a query and target sequence pair with two k-mer matches found on the same diagonal, an ungapped, and finally a gapped alignment, using Smith-Waterman [226], is performed [223].

While prefiltering greatly speeds up sequence search algorithms, it can cause issues when searching a set of closely related sequences, as is the case with antibodies, as the prefiltering step can remove good hits. Further, each tool uses an alignment method designed for general protein sequences, which can result in unreliable antibody alignments, especially in the highly variable CDRs. Within the immunoinformatics field, this alignment problem is often overcome by using antibody specific numbering schemes, like the ImMunoGeneTics (IMGT) scheme [30, 220]. Another issue with non-antibody specific tools is the lack of flexibility in their searches. These tools can only readily be used for searching against the whole antibody chain of target sequences and not for finding similar sequences based on specific subregions. Searching for identical regions at specific antibody positions, especially the CDRs, is often used when looking for similar binders [73]. With the majority of the residues involved in binding being located in the CDRs, the sequence identity over these regions is often more relevant than that of the whole antibody. For some applications, the exact set of residues involved in the paratope may be known. In these cases, searching based on the sequence identity of the paratope may be even more informative (see Figure 4.1C). An antibody specific tool that utilizes antibody numbering schemes for better searches, without prefiltering for an exhaustive search, and with the ability to search user-defined continuous or non-continuous regions, such as the paratope, would improve our ability to make best use of the antibody sequence data available.

Recent efforts to create antibody specific searching tools include iReceptor [66], AbDiver [227] and CompAIRR [228]. iReceptor, only allows for a V-, D-, or J-gene search or an exact CDR3 match search. AbDiver uses an antibody numbering scheme to align sequences and allows for both CDR3 and whole V domain searches. AbDiver restricts CDR3 searches against CDR3s with a specified V gene and species of origin, and whole V domain searches against sequences with same length CDR1 and 2 and ± 1 length CDR3. These restrictions narrow and greatly speed up the search but can occasionally lead to it finding no matches. Further, both iReceptor and AbDiver are not open-source and are only freely available to use via their

website, so can only be used against their own databases. While CompAIRR is designed for finding the overlap of CDR3's across different antibody repertoires, it can also only be used to search for either exact or similar CDR3's. However, like iReceptor and AbDiver, the restriction of the search limits its use cases, for example none of the tools can search for exact or similar CDR1 or 2, or combinations of CDRs. There therefore exists the need for an open-source antibody specific tool not limited by either being low-throughput, non-exhaustive, or only searching against entire V domain sequences.

Here, I introduce Known Antibody Search (KA-Search), a tool that allows for rapid amino acid sequence identity search across the VH and VL domains of billions of unpaired antibody chains, across either the whole domain, the CDRs, or a user defined antibody region. We demonstrate KA-Search can be used to find the most similar sequences from the ~ 2 billion heavy chain sequences in the OAS database within 30 minutes using 5 CPUs. We also show how KA-Search can be used for immune repertoire mining to obtain new insights about an antibody of interest. KA-Search is freely available at <https://github.com/oxpig/kasearch>.

4.3 Method

4.3.1 Data preprocessing

KA-Search pre-aligns the V domain of antibody sequences to a canonical alignment capable of accommodating the most common numbering positions. To do this, every sequence is first numbered with ANARCI [37] using the IMGT numbering scheme [30, 220] and then converted to a vector of the same length. As our canonical alignment, we use all of the 196 unique positions seen in at least 40,000 different sequences in OAS, as of May 2022, and four additional unique positions seen in therapeutics from Thera-SAbDab [229]. The exact unique positions are given in Table 4.1, and cover around $\sim 99.8\%$ of sequences in OAS. The 0.2% of antibody sequences that contain a rare insertion in their V domain cannot be searched using KA-Search. This set of unusual sequences is provided together with the aligned

sequences and can be searched using other methods. Every aligned sequence is accompanied by two index values which can be used to retrieve its metadata.

The KA-Search canonical alignment's unique positions	
FRW1	1, 2, 3, <u>3A</u> , 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26
CDR1	27, 28, 29, 30, 31, 32, 32A, 32B, 33C, 33B, 33A, 33, 34, 35, 36, 37, 38
FRW2	39, 40, 40A, 41, 42, 43, 44, 44A, 45, 45A, 46, 46A, 47, 47A, 48, 48A, 48B, 49, 49A, 50, 51, <u>51A</u> , 52, 53, 54, 55
CDR2	56, 57, 58, 59, 60, 60A, 60B, 60C, 60D, 61E, 61D, 61C, 61B, 61A, 61, 62, 63, 64, 65
FRW3	66, 67, 67A, 67B, 68, 68A, 68B, 69, 69A, 69B, 70, 71, 71A, 71B, 72, 73, 73A, 73B, 74, 75, 76, 77, 78, 79, 80, 80A, 81, 81A, 81B, 81C, 82, 82A, 83, 83A, 83B, 84, 85, 85A, 85B, <u>85C</u> , <u>85D</u> , 86, 86A, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 96A, 97, 98, 99, 100, 101, 102, 103, 104
CDR3	105, 106, 107, 108, 109, 110, 111, 111A, 111B, 111C, 111D, 111E, 111F, 111G, 111H, 111I, 111J, 111K, 111L, 112L, 112K, 112J, 112I, 112H, 112G, 112F, 112E, 112D, 112C, 112B, 112A, 112, 113, 114, 115, 116, 117
FRW4	118, 119, 119A, 120, 121, 122, 123, 124, 125, 126, 127, 128

Table 4.1: Overview of the 200 unique positions in our canonical alignment. The positions are based on IMGT numbering of the variable domain [30, 220], however, instead of representing CDR3 gaps with numbers (i.e. 112.1) I use letters (i.e. 112A). We choose all 196 unique positions seen in at least 40.000 different sequences in OAS, as of May 2022, and four additional unique positions seen in therapeutics from Thera-SAbDab [229]. The four additional positions are 3A, 51A, 85C and 85D (underlined).

All amino acid sequences of the antibody VH and VL domains (derived from the `sequence_alignment_aa` column) in OAS (September 2022) are pre-aligned using this method to generate a dataset ready to be used by KA-Search. This took around 2 days on 30 CPUs (starting from the already numbered sequences in OAS) and resulted in over 2,070 million heavy and 355 million light chain sequences. Sequences are split into heavy and light chains, and by species information, e.g. human, mouse, rabbit, rat, rhesus, camel and humanised, allowing for faster specific searches. We call this data set of heavy and light chains OAS-aligned. OAS-aligned also contains sequences that cannot be aligned in files labelled as unusual, for search using other methods. We also built a subset of the heavy chain dataset, OAS-aligned-small, that contains 118 million heavy chain sequences, which was generated by removing sequences containing ambiguous residues or seen less than

five times. Further, a smaller subset of 10 million human heavy chain sequences, OAS-aligned-tiny, was built by removing any sequence in OAS-aligned-small not having a residue at position one and removing duplicate sequences. OAS-aligned, OAS-aligned-small and OAS-aligned-tiny, and the code to update the data sets or expand it with an in-house data set is made freely available with KA-Search (<https://github.com/oxpig/kasearch>).

4.3.2 Identity calculation

The identity between a region in the query and target sequence is computed as the percentage of identical residues across a specific region, including insertions and deletions present in only one of the sequences. A region can be either the whole V domain or a set of antibody numbering positions, such as the CDR3. KA-Search can calculate either length matched or non-length matched sequence identity. Length matched sequence identity is only calculated if the compared region has the same length in both the query and target sequence, while non-length matched sequence identity will be calculated for all sequences. The identity can also be calculated excluding missing residues at the ends of the sequences; however, the default is to include them. By converting the query and target sequences into fixed length vectors, their sequence identity can be calculated using matrix operations, such as multiplication or equality. For KA-Search, this is implemented using the heavily optimised library JAX [230].

To search for the identity of a specific user-defined region, i.e. the CDRs, a list with the desired positions can be specified (see Figure 4.2). These positions need to be one of the 200 unique positions in the canonical alignment. In default mode, KA-Search will search for similar whole V domains of variable length, and the three CDRs and CDR3 regions with exact length match. KA-Search returns for each target sequence, the sequence identity of the defined region and the target sequence's metadata, sorted by sequence identity. For the OAS derived data, the metadata includes each column from AIRR's rearrangement schema [231]

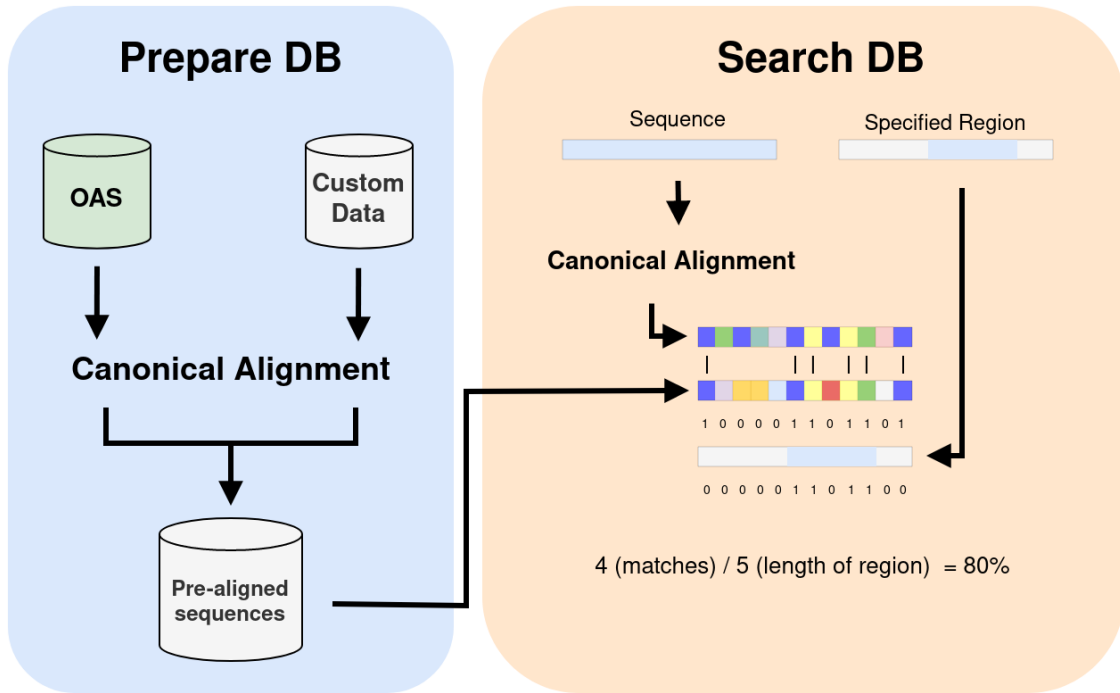


Figure 4.2: Overview of KA-Search. Before search, target sequences are pre-aligned using a canonical alignment of 200 unique antibody positions. Once an antibody has been entered into a pre-aligned database this calculation does not have to be repeated even when new data is added. For search, a query and the specific region to search for, e.g. whole variable domain or CDRs, is specified. The query is then aligned using the canonical alignment and matched with each aligned target sequence. The specific region mask is then applied before calculating the sequence identity for the region. The exact method for sequence identity calculation is either with or without length match.

and additional columns derived when preparing OAS [5] with the last column being the sequence identity.

4.3.3 Sensitivity and speed comparison

KA-Search was compared to BLASTp (version 2.13.0), CD-Hit-2d (version 4.8.1) and MMseqs2 (version 13.45111), for sensitivity and speed at searching for the closest whole antibody chain match. A set of 100 randomly selected non-redundant heavy chains of therapeutics were used to search for the closest sequence within OAS-test, a set of 10 million heavy chain antibody sequences that could be aligned with the KA-Search canonical alignment. The sequences in OAS-test were randomly extracted from the full OAS database, cleaned and reduced as done in [67]. The 100 therapeutic heavy chains and OAS-test are available for download, see Data Availability.

For BLASTp, we first pre-built a BLAST database of OAS-test using makeblastdb. BLASTp was then run using an expect value threshold of 10, word size of 3, BLOSUM62 as the substitution matrix, gap costs of 11 for existence and 1 for extension, with conditional compositional score matrix adjustment and no other filters or masks. For sensitivity comparisons we sorted hits by the expect value. With CD-Hit we searched using a sequence identity threshold of 70% for the global sequence identity and a word length of 5. For MMseqs2, we first pre-computed a sequence database of OAS-test with the createdb and createindex modules. The prepared database was then searched using the easy-search workflow with the default arguments; sensitivity of 5.7, BLOSUM62 as the substitution matrix, gap costs of 11 for existence and 1 for extension. For sensitivity comparison, 300 sequences were allowed to pass prefiltering and were thereafter sorted by sequence identity. Lastly, for KA-Search we used a pre-aligned OAS-test aligned as described above. For sensitivity comparisons, we compared over the whole V domain without length matching. For each tool, speed was calculated using the same single CPU to search for one sequence against OAS-test and sensitivity by how well each tool found the closest sequence in OAS-test to each query. The closest sequence was defined by either having the highest KA-Search identity or the highest BLOSUM62 score among the Top-100 returned from each method. The BLOSUM62 score was calculated after alignment with the Smith-Waterman algorithm, using BLOSUM62 as the substitution matrix, gap costs of 11 for existence and 1 for extension. All time measurements were performed using CPUs from an Intel Xeon Gold 6240 Processor.

4.4 Results

4.4.1 Computational speed of KA-Search

KA-Search's exact speed is dependent on the hardware used, the number of queries, number of output sequences desired and number of regions searched over.

Figure 4.3 shows a comparison between different KA-Search runs with different numbers of CPUs, when searching against the 2,070 million heavy chains in OAS-aligned. The number of closest matches returned has a minimal impact on speed,

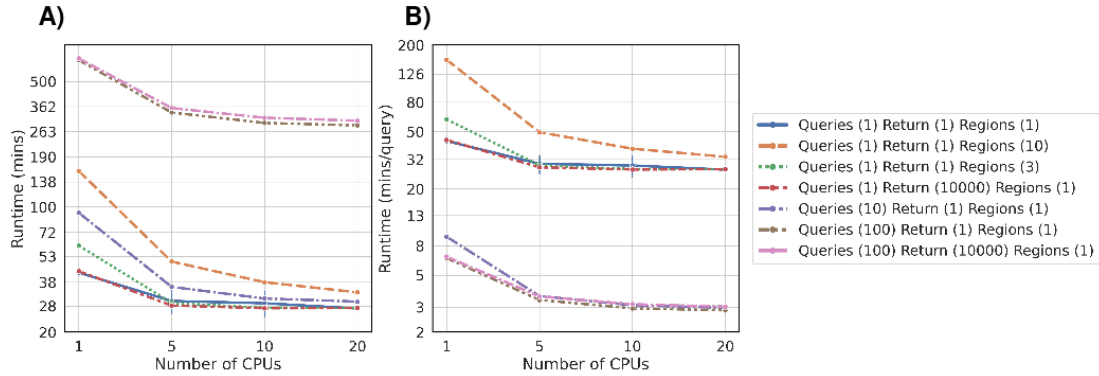


Figure 4.3: Runtime comparison of KA-Search with different numbers of queries, returned sequences and number of regions. Each search was done against the 2,070 million heavy chains in the Observed Antibody Space database. **A)** The runtime is minimally impacted by the number of closest matches returned, and increases when searching over multiple regions simultaneously or with multiple queries. **B)** Searching with multiple queries greatly reduces the runtime per query, up until 10 queries per search. It is therefore optimal to search with many queries simultaneously.

with returning the best or 10,000 best matches taking approximately the same time. Searching over multiple regions simultaneously slows the search but is faster than doing them individually. When using a single CPU one region takes $43.01\text{min} \pm 9\text{s}$, three $60.85\text{min} \pm 13\text{s}$, and ten $158.45\text{min} \pm 2\text{s}$. The time required per query is reduced when searching with multiple queries at a time, as searching with a single query takes ~ 43 minutes while searching with 100 queries takes ~ 6.6 minutes per query. KA-Search is limited by loading data into memory when searching with few queries. The optimal use of KA-Search is therefore to search with many queries and multiple regions simultaneously using multiple CPUs.

4.4.2 Comparison with other common sequence identity search tools

To compare KA-Search with current freely available and downloadable protein sequence search tools, we selected the amino acid sequence of the VH domain for 100 non-redundant heavy chains of therapeutics, and searched for the most similar sequence within OAS-test, a set of 10 million VH antibody sequences (see methods), using BLASTp [221], CD-HIT-2D [222], MMseqs2 [223] and KA-Search. For each tool, the mean and standard deviation of their speed was calculated based on seven

runs (see Figure 4.4A). KA-Search takes $8.3s \pm 22.7ms$, which is far faster than BLASTp and CD-HIT-2D, $103s \pm 75.9ms$ and $82s \pm 55ms$ seconds respectively, but slower than MMseqs2 at $3.57s \pm 78.8ms$.

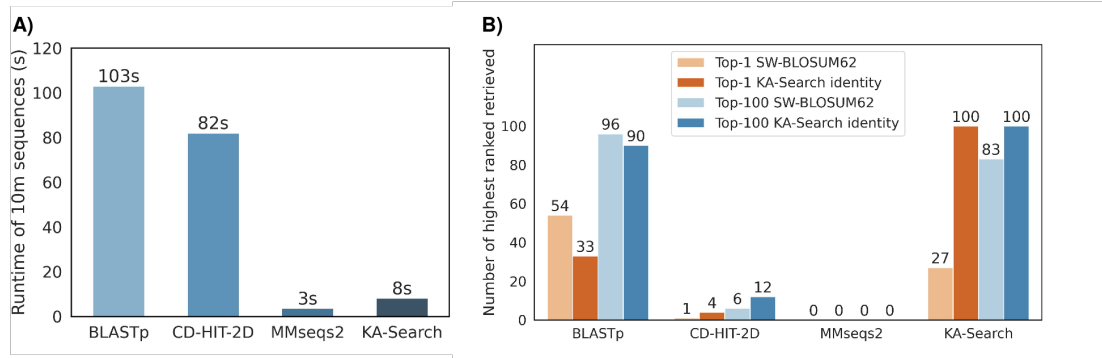


Figure 4.4: Speed and sensitivity comparison between KA-Search and the commonly used protein search tools BLASTp (version 2.13.0), CD-Hit-2d (version 4.8.1) and MMseqs2 (version 13.45111). The default setting was used for each tool. **A)** shows runtime calculated as the average time it took to search 10 million sequences (OAS-test) with a single query on a single CPU and **B)** shows sensitivity calculated by how often each tool returned the closest or the closest within the top-100 match for 100 heavy variable domains against the same 10 million sequences. The closest match was defined using two different identity metrics, the highest BLOSUM62 score after alignment with Smith-Waterman and the KA-Search identity.

In terms of sensitivity, we examined the ability of these sequence search methods to identify the most similar antibody sequence in OAS-test as defined by either the Smith-Waterman aligned BLOSUM62 score used by BLASTp, or the KA-Search identity (see Figure 4.4B). Tools that use prefiltering struggle to find the exact closest match. CD-HIT-2D's highest ranked sequences matched the target sequence with the best BLOSUM62 score and KA-Search identity for only one and four out of the 100 sequences, respectively, and MMseqs2's highest ranked sequences matched none of the closest target sequences for either metric. When looking for the closest match within the top-100 highest ranked sequences, CD-HIT-2D found the closest match based on the BLOSUM62 score and KA-Search identity for 6 and 12 sequences, respectively, while MMseqs2 found none. BLASTp and KA-Search find the closest match as highest ranked based on the BLOSUM62 score for 54 and 27 sequences, respectively, and for 33 and 100 sequences based on the KA-Search identity. Within the top-100 highest ranked sequences, BLASTp and

KA-Search find the closest match for 96 and 83 sequences, respectively, and 90 and 100 sequences based on the KA-Search identity.

We further compared the sensitivity of each method by calculating how similar the returned sequences are to the query. This comparison is shown in a density plot of the difference in sequence identity, based on exact matches, between the query and highest ranked sequences (see Figure 4.5). For BLASTp, CD-HIT-2D, MMseqs2 and KA-Search this difference was on average 15.26%, 23.04%, 20.29% and 14.43% identity, respectively. The difference between the query and the best within the top-100 highest ranked sequences were 14.12%, 21.41%, 19.52% and 14.06%, respectively.

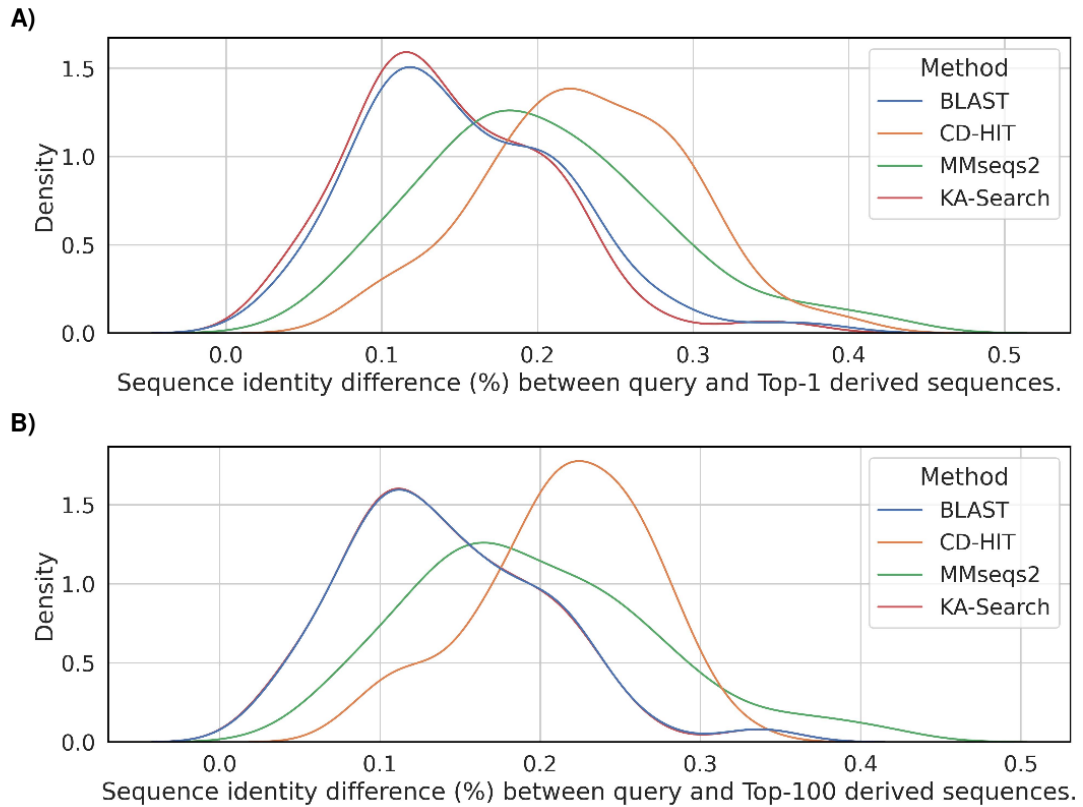


Figure 4.5: Comparison of the density of sequence identity difference between the 100 test queries and **A)** the Top-1 returned sequences and **B)** the closest within the Top-100 returned sequences from KA-Search and the commonly used protein search tools BLASTp (version 2.13.0) [221], CD-Hit-2d (version 4.8.1) [222] and MMseqs2 (version 13.45111) [223]. The default setting was used for each tool. Sequence identity was calculated as the percentage of exact matches after alignment with the Smith-Waterman algorithm using BLOSUM62 [224] as the substitution matrix and gap costs of 11 for existence and 1 for extension.

While CD-HIT-2D is better than MMseqs2 at finding the closest match, MMseqs2 returns on average better matches. The highest ranked from BLASTp and KA-Search are slightly biased towards their used metric; however, the closest match within the Top-100 from both methods, are very similar. Unlike all the other methods KA-Search is exhaustive, so it finds the exact closest match every time using the KA-Search identity.

4.4.3 Immune repertoire mining with the COVOX-253 antibody

COVOX-253 is an antibody which binds to the neck of SARS-CoV-2's Receptor-Binding Domain (RBD) [232]. Using KA-Search, up to the 1,000 closest sequences to the heavy chain of COVOX-253, with over 90% identity, were extracted for four different regions: the whole V domain, the three CDRs, the CDR3 and the paratope. The paratope was derived from the PDB structure 7BEN and defined as any residue in the antibody which was within 4.5Å of the RBD [170]. Figure 4.6A shows the disease of the patient the antibody sequence found in OAS comes from and in Figure 4.6B each antibody's combination of V and J genes. Most matched sequences are derived from the gene alleles IGHV1-58*01 and IGHJ3*02; however, COVOX-253's CDR3 is seen with six different V gene alleles, IGHV1-58*01, IGHV1-58*02, IGHV1-18*01, IGHV1-46*01, IGHV1-69*10 and IGHV1-69*13.

Searching for the closest match using the whole V domain returns 822 sequences from healthy individuals and 178 from patients with one of nine different diseases, eight which are SARS-CoV-2. Searching with the CDR positions returns one sequence from a healthy individual and 789 sequences from patients with SARS-CoV-2, while searching with the CDR3 or paratope positions returns 197 and 124 sequences, respectively, all from SARS-CoV-2 infected patients. The fact that OAS-aligned only contains ~84 million heavy chains from patients with SARS-CoV-2, ~4% of all heavy chains in OAS-aligned, highlights the importance of being able to search over specific regions.

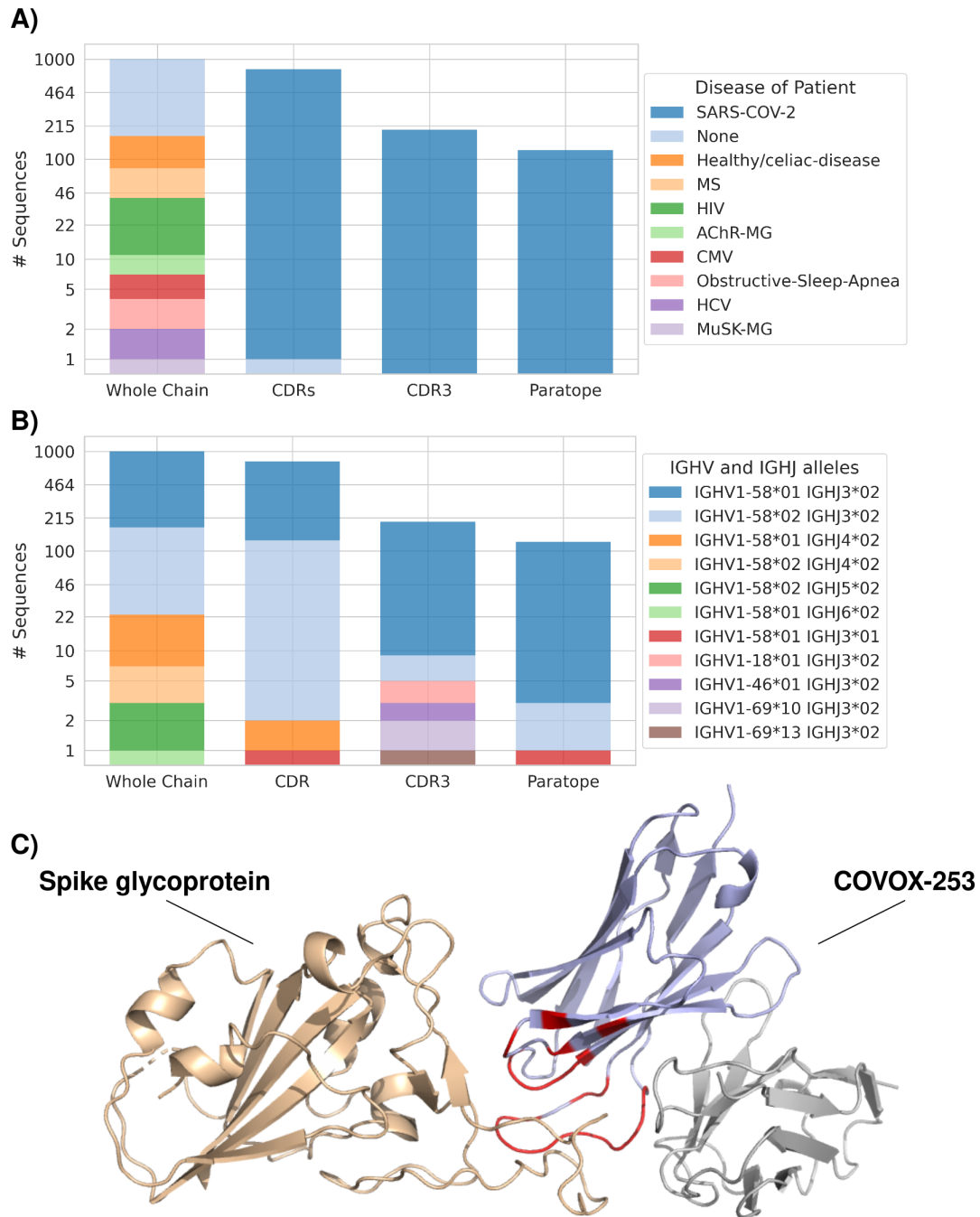


Figure 4.6: A KA-Search for sequence matches to the heavy variable domain of the SARS-CoV-2 RBD binding antibody COVOX-253. Returned antibodies with over 90% identity across four different regions were visualised based on, **A)** the disease state of the patient and **B)** which V- and J-genes the antibody sequence is derived from. The y-axis is scaled logarithmically to better visualise the data. **C)** The variable domain of COVOX-253's heavy (purple) and light (grey) chain with the bound spike glycoprotein (beige), derived from PDB 7BEN. The paratope of the heavy chain, which was used to search with KA-Search, is shown in red.

4.5 Discussion

Immune repertoire mining is a powerful method for identifying antibodies in nature which are similar to an antibody of interest and can help indicate likely specificity or immunogenicity. However, the number of available antibody sequences are now in the billions and is continuing to grow. Therefore, repertoire mining for highly similar sequences has become increasingly computationally expensive. Existing approaches are limited by either being low-throughput, inaccessible for large scale searches, non-exhaustive, not antibody specific, or only searching against entire V domain sequences. There is therefore a need for a specialised tool, optimised for a rapid and exhaustive search of any antibody region against all known antibodies, to better utilise the full number of available repertoire sequences.

In this chapter, I introduce Known Antibody Search (KA-Search), a platform independent antibody search tool. KA-Search finds antibody sequences with an accuracy comparable to BLASTp, while being over an order of magnitude faster and allowing searches over specific regions. KA-Search exploits antibody numbering, allowing us to pre-align antibody sequences to a fixed-length vector. This circumvents pairwise alignment during search, an otherwise time-consuming step. This was done to keep the alignment short and increase speed. The increased speed allows KA-Search to avoid prefiltering and be exhaustive while still retaining a competitive speed. Avoiding prefiltering is crucial, as current prefiltering techniques greatly reduce sensitivity when searching highly related proteins, such as antibodies, where a single mutation can be of great importance. While pre-aligning the antibody sequences increases search speed, the initial pre-alignment is slow. We therefore provide a pre-aligned dataset of the current OAS, ready to use for searching. This dataset can be extended with future OAS updates or in-house data without the need to re-align the existing sequences.

Pre-aligning sequences also opens new use-cases. Instead of only searching against the whole antibody V domain, searches can now be focused on specific positions in the alignment. Searches can be specific for the CDRs or regions specific for individual antibodies, such as the paratope. This flexibility allows for studies

which were previously difficult to execute. As an example, previously an extensive study was needed to search OAS across the whole V domain, CDRs and CDR3 for the closest match to a set of 242 therapeutics [212]. The same study can now be done on the 804 therapeutics within Thera-SAbDab [229] (as of August 2022) with KA-Search in less than two days compute and little configuration (see Supplementary Figure S4 of the KA-Search paper [4]). KA-Search can also extract the metadata from OAS for the matched sequences, which can be used to obtain new insights about an antibody of interest. Using KA-Search to find the closest sequences with or above 90% identity across four different regions for the SARS-CoV-2 RBD binding COVOX-253 demonstrates the power of searching across particular regions. The closest matches from searching with the whole V domain comes from healthy patients or patients with a variety of diseases. However, binding region specific searches only return sequences found in SARS-CoV-2 infected patients. These sequences could therefore also have RBD binding properties and are possible candidates for further affinity studies. Investigating the genes of the closest sequences, also potentially indicates which other frameworks a region of interest could exist on. For COVOX-253, the CDR3 is seen in sequences with six different V genes, which each could be possible framework candidates for the CDR3. The ability to return high numbers of close matches without decreasing speed, also opens up KA-Search as a means for creating multiple sequence alignments of similar antibody sequences.

A limitation of KA-Search, is that the current version only searches across the V domain, disregarding the constant domain. This is driven by the current very limited number of available sequences of the constant domain. KA-Search can also only search with and against sequences which can be numbered and aligned with the 200 unique positions in the canonical alignment described in the methods. The 0.2% of sequences with rare insertions cannot be searched with KA-Search and are excluded from OAS-aligned. The rare insertions are seen across the whole V domain and while most are likely derived from sequencing or ANARCI numbering errors, e.g. the highly unlikely eight residue insertion giving position 81I, some rare insertions can be contributed to limited data of certain species within OAS, e.g.

position 112M seen in camel sequences with a CDR3 longer than 37 residues. As currently OAS mainly contains human and mouse sequences, the 200 positions cover those species well but may as described cover less of the sequences derived from other species. With OAS growing, the unique positions can be updated in the future to better handle other species. Sequences that failed to be aligned are also provided in the OAS-aligned download and can be searched using other methods if desired, for example if the sequence of interest contains rare insertions. Further, KA-Search currently only finds similar sequences using sequence identity. While this is sufficient for finding sequences with few mutations, calculating the sequence similarities using a substitution matrix would allow the exploration of more distant matches.

KA-Search can be run on any system with Python, requiring only 6GB RAM for searching OAS-aligned and 2GB RAM for OAS-aligned-tiny. This enables any researcher to readily search for similar sequences. Currently, when searching with few queries, the primary bottleneck related to the speed is the loading of data into memory to search against. For optimal use or large scale studies, KA-Search therefore benefits considerably from searching with many queries simultaneously on a high-performance computer using multiple CPUs.

KA-Search's speed, exhaustiveness and flexibility allows it to search the vast numbers of antibody sequences now available seamlessly, find viable mutations and gain new insight into antibodies of interest. I therefore believe KA-Search is a useful tool that will allow the antibody community to explore antibodies in new ways. To maximize KA-Search's possible contribution to the community, KA-Search is open source and freely available at <https://github.com/oxpig/kasearch>.

Although mining repertoire data can be useful for identifying natural antibodies with similar properties, the amount of data obtained in each next generation sequencing experiment makes it hard to discern what the function of each individual sequence is. In Chapter 5 I describe PLabDAb, a large database of functionally annotated paired antibody sequences and structural models from papers and patents.

5

PLAbDab

Contents

5.1	Chapter abstract	86
5.2	Background	87
5.3	Methods	88
5.3.1	Collecting unpaired antibody sequences	88
5.3.2	Creating paired antibody sequences from the unpaired data	88
5.3.3	Searching PLaBdab	90
5.4	Results	90
5.4.1	Database statistics	90
5.4.2	Searching PLaBdab	92
5.4.3	Using PLaBdab to Generate New Datasets	96
5.5	Discussion	96

5.1 Chapter abstract

This chapter is based on the paper “The Patent and Literature Antibody Database (PLaBdab): an evolving reference set of functionally diverse, literature-annotated antibody sequences and structures” [6]. It describes PLaBdab, a self-updating repository containing over 150,000 paired antibody sequences and 3D structural models, of which over 65,000 are unique. Each entry in the database also contains the title and authors of its literature source. I describe the methods used to extract,

filter, pair, and model the antibodies in PLaBdab, and showcase how PLaBdab can be searched by sequence, structure, or keyword.

PLaBdab uses include annotating query antibodies with potential antigen information from similar entries, analysing structural models of existing antibodies to identify modifications that could improve their properties, and compiling bespoke datasets of antibody sequences/structures known to bind to a specific antigen. PLaBdab is freely available via Github (<https://github.com/oxpig/PLaBdab>) and as a searchable webserver (<https://opig.stats.ox.ac.uk/webapps/plabdab/>).

5.2 Background

As described in Chapter 1, antibodies are by far the most successful type of biotherapeutic, with over 100 approved by the FDA and many more in the advanced stages of clinical development [13, 229] (see Section 1.2.2). Their high specificity and affinity also make them a valuable tool in many areas of medical and scientific research. For example, antibodies are used routinely in diagnostic assays [233], and to better understand the effects of vaccination on the immune system [234].

Next generation sequencing (NGS) of the immune repertoire has lead to the generation of vast amounts of antibody sequence data (see Section 1.2.6). Although NGS data has proven invaluable to compare repertoires between individuals, it provides little information on the functions of individual sequences within a repertoire. However, there also exists a large number of smaller scale studies, each one dedicated to investigating a small number of antibodies. When combined, the antibodies from these studies amount to a large number of sequences with rich metadata. There are a number of databases that aim to compile subsets of this data, for example SAbDab [154, 156] for antibodies with resolved crystal structures, Thera-SAbDab for antibody therapeutics [229], CoV-AbDab for COVID-19 binding antibodies [235], or PAD for unpaired antibody sequences from patents [236]. Paired antibody sequences with information on their epitope can also be obtained from IEDB [237].

In this chapter I present PLaBdab, a database containing 150,000 paired antibody sequences from over 10,000 small scale studies. PLaBdab is larger than any other non-NGS database of paired antibody sequences by at least an order of magnitude. We make the data freely available and provide methods to rapidly search it by either sequence identity using KA-search [4], structural similarity [160, 161], or by keywords in the title of the study. Each of the sequences comes with a direct link to its source material, making it easy to obtain additional information about any antibody of interest.

5.3 Methods

5.3.1 Collecting unpaired antibody sequences

The majority of data in PLaBdab is extracted from the Protein database of NCBI [238]. The BioPython Entrez module [239] is used to query the database for entries containing the words “antibody”, “antibodies”, “immunoglobulin”, “scfv” or “bcr” in any of their fields. Entries with sequences longer than 1000 amino acids or shorter than 70 are removed at this stage. Entries containing the words nanobody or nanobodies are also removed. Around 2.5 million entries were returned from this search.

Sequences for each of these 2.5 million entries were then searched for antibody variable domain sequences using ANARCI [37]. This resulted in around 530,000 potential antibody variable domain sequences from around 13,000 different sources.

5.3.2 Creating paired antibody sequences from the unpaired data

For a large number of entries the metadata provides enough information to pair the heavy and light variable domains. For entries from the same literature source, VH-VL pairing was attempted using the following heuristics:

1. If there is only one VH and only one VL within a single entry, these were paired together.

2. If there is a unique non-common word that is found in the description of exactly one VH and one VL, these are paired together. Uncommon words are defined as words found in the description of less than 20 entries in the unpaired database.
3. Some entries contain information on the experimental source, such as the isolate or clone. If there is a unique VH and VL from the same experimental source they are paired together.
4. Entries from patents have sequence IDs by which they are referred to in the patent text. If the patent text mentions a VH and VL within the same paragraph, these are paired together. If a paragraph in the patent text mentions the same number of VH and VL entries, these are paired in the same order as they are mentioned.
5. If there is a single VH (or VL) from one source, this entry is paired with all other VL (or VH) entries from that source.
6. If there are the same number of VH and VL entries from one source, they are paired in order.

The described strategies for pairing VH-VL sequences have varying levels of accuracy. Twenty entries paired by each of the methods described above were randomly selected and manually checked to estimate the accuracy of each pairing method. Methods 1-4 achieved perfect accuracy on the twenty entry test set. Method 5 incorrectly paired one entry and Method 6 incorrectly paired two entries. Each paired entry is given a flag to indicate how it was paired, making it easy for users to filter for less accurate pairing methods. To further increase the coverage of the dataset, sequences from both SAbDab and Thera-SAbDab were also added to PLaBdab.

5.3.3 Searching PLaBdab

To allow users to rapidly search the database for antibodies with a similar sequence, we implemented KA-Search [4]. KA-search is able to carry out rapid and exhaustive sequence identity searches, allowing users to find similar sequences over the whole variable domain, the CDR loops, the CDR-H3 or a user defined region. The entire database can be queried with KA-Search in under 5 seconds on 5 CPUs.

Structurally similar antibodies can have similar functions, even if they are distantly related in sequence [160, 161]. To enable CDR structure-based searching, we model all paired sequences in the database using ABodyBuilder2 [2]. Entries missing more than eight residues at the start of the sequence were restored using AbLang [67] prior to modelling. Entries with the same sequence were only modelled once, in total 65,000 unique antibody models were generated.

To structurally search PLaBdab, query sequences are first modelled using ABodyBuilder2, without refinement. The refinement step was skipped in order to boost speed with minimal compromise on backbone model quality. The framework of the predicted structure is then aligned to the structure of all other entries in PLaBdab with the same CDR loop lengths. Lastly, the carbon-alpha (C_α) root-mean squared deviation (RMSD) over all CDR residues is computed and used to rank entries. Searching the database in this way takes around 10 seconds on 5 CPUs.

Finally, as each sequence in PLaBdab is linked to a patent or publication, I provide users the ability to search fields, such as the title of the study, using regular expressions.

5.4 Results

5.4.1 Database statistics

Currently the total number of entries in PLaBdab is around 150,000, over 90% of which are paired with high confidence using methods 1-4, or comes from therapeutic or crystal structure entries. As can be seen in Figure 5.1A, the number of antibody sequences that could be collected by the PLaBdab methodology has been steadily

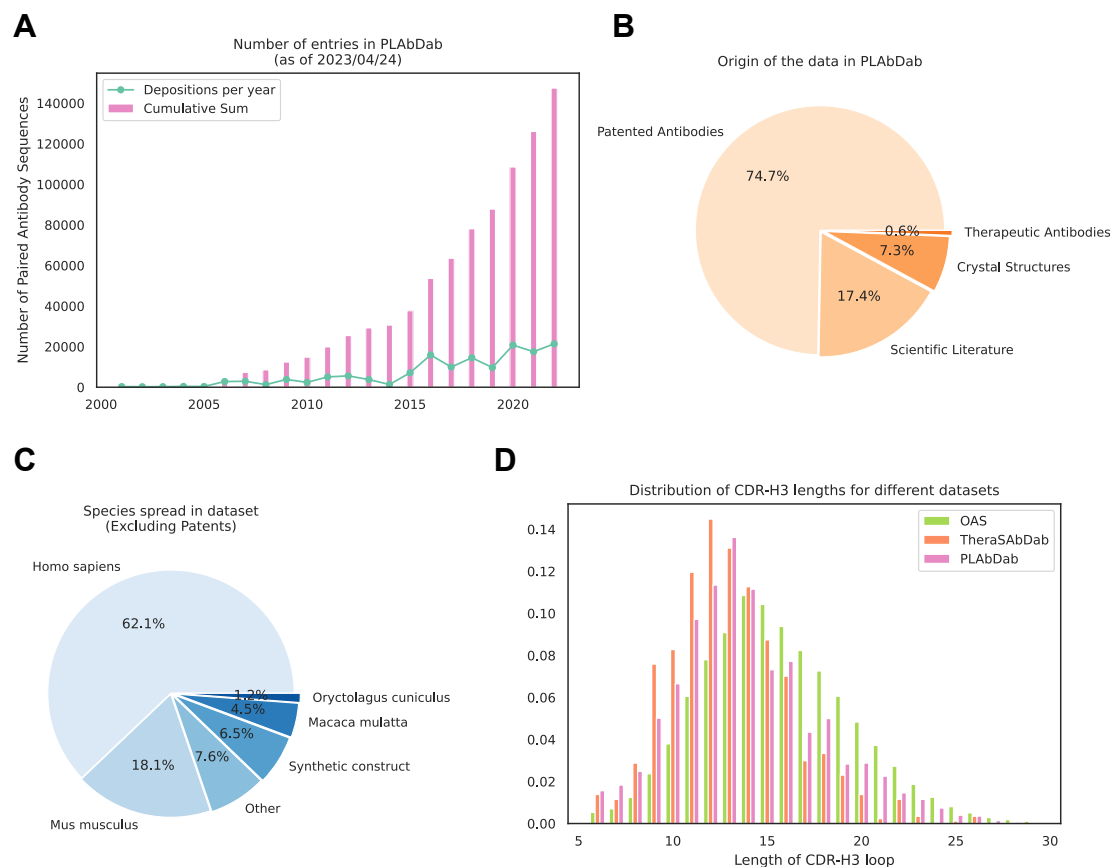


Figure 5.1: Summary statistics for PLaBdab. (A) Number of new antibody sequences each year and total number of antibody sequences that could be collected by the PLaBdab methodology. (B) The proportion of data in PLaBdab by type of source. (C) The proportion of antibodies from each species in non-patent entries. (D) Comparison of the distribution of CDR-H3 lengths in PLaBdab, Thera-SAbDab [229], and OAS [5].

growing since the early 2000s, with somewhere between 10,000 and 30,000 new antibody sequences being published each year for the last five years.

Figure 5.1B shows the distribution of PLaBdab entries by source, around three quarters of entries come from antibody sequences described in patents, while less than 20% are derived from the scientific literature. This may be due to patents following a more standardised way of depositing antibody sequences. The rest of the entries in PLaBdab come either from solved structures in SAbDab [154], or from additional sequences in Thera-SAbDab [229].

Patent applicants are not obligated to upload species information about their sequences to NCBI. This means that the majority of entries in PLaBdab lack a

species annotation. However, species information does exist for the majority of the entries sourced from the academic literature. Across the sequences with species information, most entries are labelled as human, with a smaller number of sequences annotated as mouse, macaque or rabbit. (Figure 5.1C).

It has been observed that therapeutic antibodies tend to have shorter CDR-H3 loops than those seen in large repertoire studies, an observation that may be due to longer CDR loops leading to issues during therapeutic antibody development [158]. The average CDR-H3 loop length from antibodies in PLaBdab (around ~14.0) falls somewhere between the average CDR-H3 loop length of OAS (around ~15.6) and Thera-SAbDab (around ~12.9) (Figure 5.1D).

5.4.2 Searching PLaBdab

Search method	Retrieved (cons.)	Sources (cons.)	Unique (cons.)
PD-1			
VH identity	576 (222)	180 (41)	258 (61)
VH+VL identity	155 (132)	39 (28)	22 (16)
CDR structure	227 (168)	60 (26)	46 (19)
CDR structure+identity	127 (127)	29 (29)	14 (14)
RSV			
VH identity	2 (2)	1 (1)	2 (2)
VH+VL identity	2 (2)	1 (1)	2 (2)
CDR structure	29 (29)	4 (4)	23 (23)
CDR structure+identity	4 (4)	1 (1)	4 (4)
CD200			
VH identity	13 (13)	4 (4)	2 (2)
VH+VL identity	13 (13)	4 (4)	2 (2)
CDR structure	1175 (95)	259 (15)	440 (50)
CDR structure+identity	37 (37)	5 (5)	5 (5)

Table 5.1: Results from searching PLaBdab with three different queries (PD1, RSV and CD200) using different methods. For each method, the number of retrieved entries, the number of different sources the entries come from and the number of unique antibodies found is given. The value given in parenthesis is the number of entries or sources that are functionally consistent with the query. Details on the three query antibodies and the four searching methods are given in the text.

PLaBdab allows users to search the database using a variety of methods. To demonstrate the sequence and structure search options, I compared the results

of searching the database in four ways:

- For antibodies with a sequence identity of over 90% over the VH (VH identity).
- For antibodies with a sequence identity of over 90% for both the VH and VL (VH+VL identity).
- For antibodies with a C_α RMSD over the CDR loops of under 1.25Å to an ABodyBuilder2 model of the query (CDR structure).
- For antibodies with a C_α RMSD over the CDR loops of under 1.25Å to an ABodyBuilder2 model of the query and sequence identity over the CDR loops of over 80% (CDR structure+identity).

I performed the above searches for three antibodies each of which target a different antigen. Antibody one binds programmed cell death protein 1 (PD-1), and was taken from the patent “Anti-PD-1 antibodies and methods of use thereof.”; antibody two binds Respiratory Syncytial Virus (RSV), sourced from the paper “Rapid profiling of RSV antibody repertoires from the memory B cells of naturally infected adult donors.” [240]; and antibody three is the therapeutic antibody Samalizumab, which binds the OX2 membrane glycoprotein (CD200). For each of the three queries I took the sequence and built an ABodyBuilder2 model for use in the structure searches. The result of using each search method described above to query PLaBdab with these antibodies is shown in Table 5.1.

The PD-1 case study shows the benefits of having the paired heavy and light chain variable domain sequences. While a sequence identity search over the VH finds antibodies binding to the same epitope around 25% (61 out of 258) of the time, including the VL improves the accuracy to around 75% (16 out of 22). The fact that there are many highly identical heavy chain sequences that bind to different epitopes suggests that, for this antibody, the light chain may contribute significantly to binding. To validate this, I analysed the crystal structure of an antibody bound to RSV which is returned by all four search methods (PDB ID: 5GGR). According to

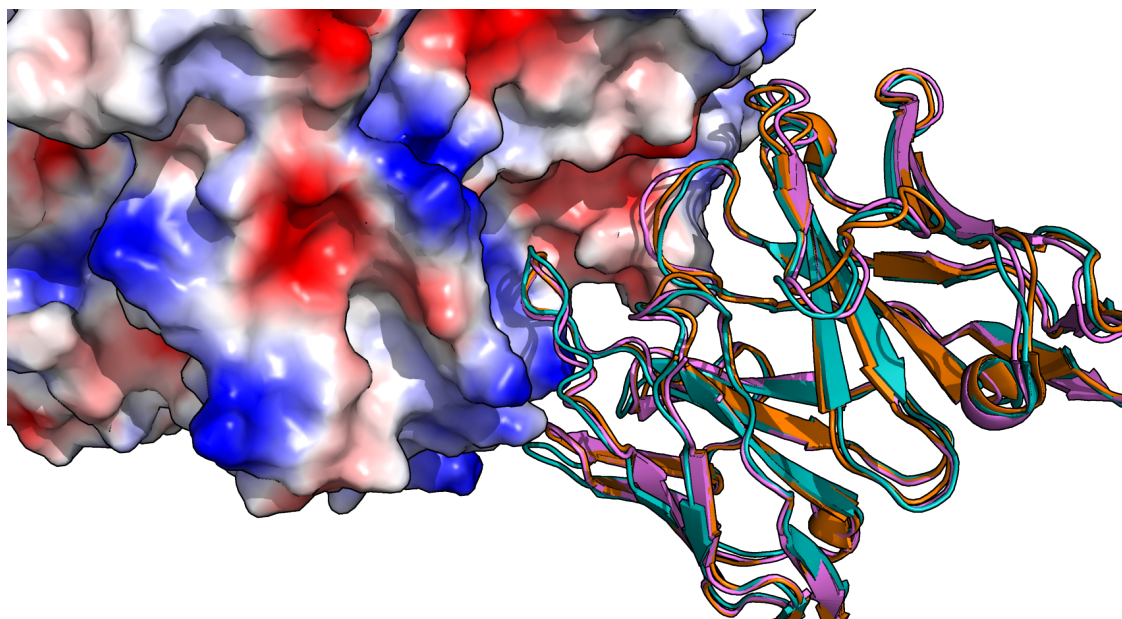


Figure 5.2: Crystal structures of anti-RSV antibodies aligned to structural model of query. A cartoon representation of the ABodyBuilder2 model for the query antibody is shown in purple with the crystal structure for 7LUC shown in blue and the crystal structure for 7LUD shown in orange. The surface of the RSV antigen is coloured based on electrostatics using PyMol [241].

Arpeggio [242], the paratope is evenly split between the heavy and light chain, with eight heavy chain and seven light chain residues found to interact with the antigen.

Searching PLaBdab for entries similar to our PD-1 target by CDR structure returns antibodies that target the same epitope around 40% of the time (19 out of 46), but many of them are from different studies than those found by sequence search. All the entries returned by a CDR structure plus sequence identity search target the same epitope as our query.

For the RSV binding antibody, there are only two entries in PLaBdab with greater than 90% sequence identity over the VH. Searching PLaBdab for entries with a similar CDR structure returns 23 unique antibodies which bind the same epitope. This suggests that this antibody may require a very specific CDR loop conformation to bind the RSV Fusion Glycoprotein [243] at that specific site. Two of the retrieved entries belong to antibodies with resolved crystal structures (PDB IDs: 7LUC and 7LUD), one of which is bound to the antigen. Figure 5.2 shows the similarity of these structures to the ABodyBuilder2 predicted structure of

the RSV binding antibody query.

Only two unique antibody sequences in PLaBdab have a VH sequence identity of over 90% to Samalizumab. A structure search using Samalizumab as a query results in over a thousand antibodies that bind a very diverse set of epitopes. In Figure 5.3 I show a selection of antibodies with very similar CDR backbone structures to Samalizumab binding to four very different epitopes on different antigens. This indicates that the backbone structure of the CDR loops is likely not the primary driver of binding specificity for this antibody. However, when adding an 80% sequence identity cutoff over the CDR loops to the structure search, we are left with five unique antibody sequences all of which bind the same epitope.

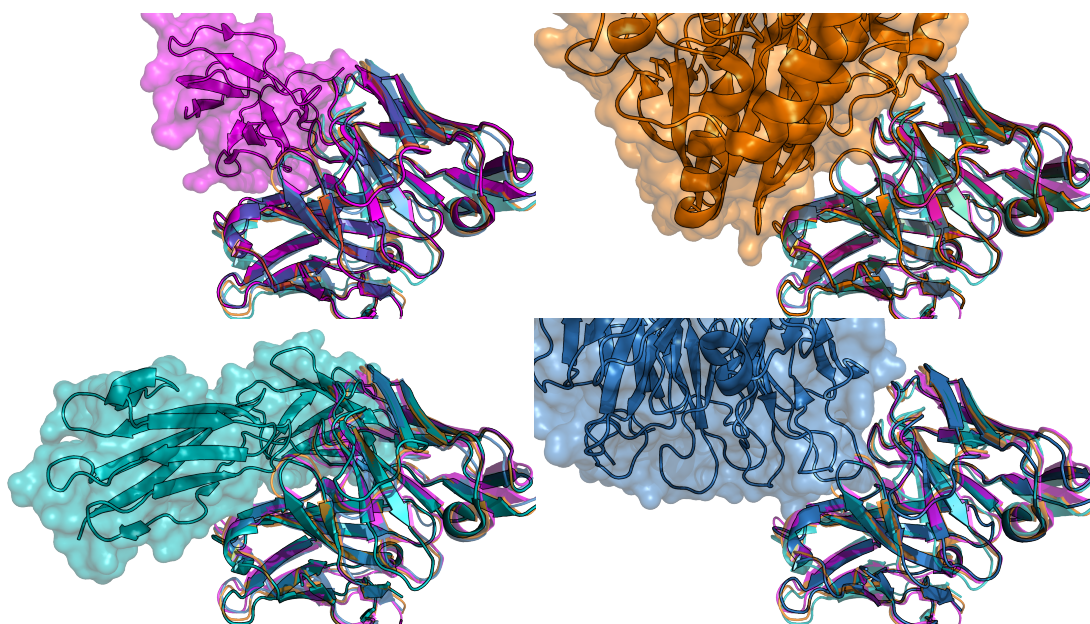


Figure 5.3: Crystal structure of four antibodies with very similar structure that bind very different antigens. The crystal structure of all four aligned antibodies is shown in each figure (4AL8 in purple, 7XY8 in green, 7LA4 in blue, and 7BBJ in orange). On the top left they are shown in complex with the antigen of 4AL8 (Dengue virus DII protein); top right 7BBJ (CD73); bottom left 7XY8 (Emmprin CD147); and bottom right 7LA4 (Integrin $\alpha_{IIb}\beta_3$).

These five case studies highlight the utility of PLaBdab’s data and various search methods to identify antibodies with similar function to a query. The case studies also highlight that any computational search should be further investigated via both the meta data in PLaBdab and, if possible, experimental means.

5.4.3 Using PLaBdab to Generate New Datasets

As PLaBdab contains a large number of antibody sequences targeting a diverse set of antigens, it can be used to generate antigen-specific antibody libraries. For example, searching PLaBdab for entries containing the keywords “ebov” or “ebola” in their title returns almost 1,500 unique antibody sequences from 56 sources. Using the keywords “hiv” or “immunodeficiency” finds over 3,800 entries from over five hundred sources, while “autoimmune” and “autoantibody” returns over 400 unique antibody sequences from over 90 sources. Not every antibody sequence retrieved from a patent or paper will necessarily bind the searched target, but such prefiltered sets provide a very valuable starting point towards generating an antigen specific library to a target of interest.

To benchmark how effective searching the titles of the sources of PLaBdab entries by keywords is at retrieving relevant antibodies, I manually inspected a random set of 100 antibodies from different studies and found that 88 are true HIV binders. Of the non-HIV binders, four were anti-idiotypic antibodies, three bound CD4, three bound other antigens and two were incorrectly paired.

5.5 Discussion

I present PLaBdab, a large database of paired antibody sequences from patents and papers. Each entry in the database contains a link to its original source material, which relates the paired sequences to useful functional information. PLaBdab can be automatically updated without any manual input, making it straightforward to keep up to date with the latest publications and patents.

To demonstrate the utility of PLaBdab, I explore different search methods for identifying antibodies with similar binding properties to a given antibody of interest. Our results demonstrate that for different antibodies, different types of search yield the best results. In all three cases investigated, CDR structure alongside CDR sequence identity gave the most accurate results, but in most cases, this also missed functionally cognate antibodies. The inclusion of the light chain in any search

always improved accuracy, in agreement with recent studies that have highlighted the importance of the light chain residue motifs for antigen binding [244], and the restricted heavy and light chain gene pairs observed among functional antibodies [80]. In one case searching by structure provides a link that is not immediately apparent from sequence search alone. This finding is consistent with recent work which uses structural information for epitope binning [160, 161].

One of the main challenges in developing PLaBdab, and the area with most potential for improvement, is the pairing of heavy and light chains based on metadata. Although the strategy described in this work accurately pairs a large number of antibody sequences, there are still many for which pairing was not possible. The PLaBdab generation code also relies on authors publishing their antibodies to databases such as NCBI [238] or the PDB [193], and although many authors do submit their sequences to these databases, there remains a large body of antibody sequence data shared in the text, figures, or tables of papers. In the future, it may also be possible to automatically extract sequence metadata from papers using large language models, such as GPT4 [245], or other natural language processing techniques.

Nevertheless, PLaBdab already contains over 60k unique annotated antibody sequences providing an invaluable resource to the antibody research community. PLaBdab can also be used to generate antigen specific libraries that could be used to train novel machine learning models or as a starting point for the development of novel therapeutics. I have made PLaBdab freely available to download and query via a webserver (<https://opig.stats.ox.ac.uk/webapps/plabdab/>) and the code is available to download via github (<https://github.com/oxpig/PLAbDab>).

In the final chapter, I provide a concise summary of the conclusions drawn from the four result chapters and the overall thesis. Additionally, I suggest potential enhancements or expansions to the tools and analysis presented in these four chapters.

6

Conclusions and Future Work

Contents

6.1	Conclusions	98
6.2	Future work	99

6.1 Conclusions

This DPhil project has made incremental improvements towards fast and accurate antibody structure prediction. I first showed the potential of equivariant graph neural networks for CDR structure prediction with ABlooper (Chapter 2). To the best of my knowledge, this marked the first application of equivariant deep learning to antibody structure prediction. However, ABlooper has some limitations: it cannot model side-chain atoms and will on occasions generate unphysical structures, which have to be corrected using computationally expensive methods.

My second iteration for antibody structure prediction, ABodyBuilder2 (Chapter 3), addressed some of these challenges. Inspired by the AlphaFold2 model, ABodyBuilder2 directly predicts side chain atoms, thus enhancing the accuracy of their modelling. By incorporating hard-coded rules for bond angles and distances within a residue, it also reduces the number of unphysical structures predicted.

ABodyBuilder2 proved comparable in accuracy to AlphaFold-Multimer while achieving over a hundred-fold increase in speed. At the time of release, it also stood as the most accurate of all antibody-specific methods. When I trained the model exclusively on T-Cell Receptors (TCRs) and nanobodies, similar results were obtained, leading to the creation of TCRBuilder2 and NanoBodyBuilder2.

Additionally, I developed two novel tools, KA-Search (Chapter 4) and PLaBdab (Chapter 5), designed to better utilise the available antibody data. KA-Search is a tool that allows for fast and exhaustive search of large NGS antibody databases by sequence identity. PLaBdab is a database that compiles antibody sequences from patents and papers and makes them searchable either by sequence or structure. I expect these tools, in combination with the improved structural modelling, to aid in the process of therapeutic antibody design.

Taken as a whole, the results presented in this thesis project signify a step forward in the fast and accurate prediction of antibody structures and the efficient use of public antibody data. The culmination of these efforts opens new horizons for accelerating therapeutic antibody design and lays the groundwork for further advances in the field of immunoinformatics.

6.2 Future work

Despite the improvements that deep learning has brought to the field of antibody structure prediction, current methods still struggle to consistently predict CDR-H3 with high accuracy. Several factors may contribute to this accuracy gap. First, the coverage provided by the currently available data may be insufficient. CDR-H3s have only been observed in a limited number of conformations which may not be sufficient for a deep learning model to discern the pattern that determines their structure. Second, some percentage of CDR-H3 loops are flexible and cannot be fully described using a single conformation. Deep learning methods could be designed to account for this flexibility by learning the distribution of possible conformations, but the amount of available data may hinder this process. Finally, current models lack awareness of the environment in which the antibody exists. For instance, an

antibody bound to an antigen may have a different CDR-H3 loop conformation than an unbound one. Strategies to incorporate antigen information into antibody modelling could potentially improve CDR-H3 structure prediction.

Deep learning models are inherently differentiable. This property enables the use of novel deep-learning based antibody structure predictors to condition generative models. For example, a model could be trained to generate antibody sequences that ABodyBuilder2 would predict to have a certain structure. This would more easily allow for understanding the effect of changing individual residues on the structure. The same model could also be used for predicting sequence-dissimilar antibodies with similar structures.

A potential extension of KA-Search would involve its use to generate a score for the naturalness of an antibody of interest. The maximum sequence identity to a human antibody repertoire could serve as an indicator of how likely a human would be to naturally generate a particular antibody. This property could be desirable when developing a therapeutic to minimise the likelihood of the antibody triggering an autoimmune response. However, for antibodies targeting certain antigens, it could indicate the evolutionary pressure the epitope is naturally under and be indicative of the likelihood of the antigen mutating at that site. Whether this property should be positively or negatively selected during therapeutic is therefore likely to be context specific.

Enhancements to PLAbDab might include methods to increase the retrieval and pairing of antibody sequences, or to convert the extracted metadata in to a more standardised format. Currently, PLAbDab can already be used for a wide range of downstream applications. One exciting unknown is how it will be used for deep learning. For instance, its ability to generate antigen-specific datasets could be used to finetune antibody generative models to target a particular antigen. As the amount and quality of data continue to increase, it may become feasible to train a model to classify antibody sequences by the antigen they target. Such a model could provide valuable insights on the diversity in function of antibodies

within an individual's repertoire. The pipeline used to generate PLabDab could also be applied to generate TCR or nanobody specific databases.

References

- [1] Brennan Abanades et al. “ABlooper: Fast accurate antibody CDR loop structure prediction with accuracy estimation”. In: *Bioinformatics* 38.7 (2022), pp. 1877–1880.
- [2] Brennan Abanades et al. “ImmuneBuilder: Deep-Learning models for predicting the structures of immune proteins”. In: *Communications Biology* 6.1 (2023), p. 575.
- [3] John Jumper et al. “Highly accurate protein structure prediction with AlphaFold”. In: *Nature* (2021), pp. 1–11.
- [4] Tobias H. Olsen et al. “KA-Search, a method for rapid and exhaustive sequence identity search of known antibodies”. In: *Scientific Reports* 13.1 (2023), p. 11612.
- [5] Tobias H Olsen, Fergus Boyles, and Charlotte M Deane. “Observed Antibody Space: A diverse database of cleaned, annotated, and translated unpaired and paired antibody sequences”. In: *Protein Science* 31.1 (2022), pp. 141–146.
- [6] Brennan Abanades et al. “The Patent and Literature Antibody Database (PLAbDab): an evolving reference set of functionally diverse, literature-annotated antibody sequences and structures”. In: *bioRxiv* (2023).
- [7] Charles. Janeway. *Immunobiology 5 : the immune system in health and disease*. English. New York: Garland Pub., 2001.
- [8] Donald N Forthal. “Functions of antibodies”. In: *Microbiology spectrum* 2.4 (2014), pp. 2–4.
- [9] Anthony R Rees. “Understanding the human antibody repertoire”. In: *MAbs* 12.1 (2020), p. 1729683.
- [10] Bryan Briney et al. “Commonality despite exceptional diversity in the baseline human antibody repertoire”. In: *Nature* 566.7744 (2019), pp. 393–397.
- [11] Nicole A Doria-Rose and M Gordon Joyce. “Strategies to guide the antibody affinity maturation process”. In: *Current opinion in virology* 11 (2015), pp. 137–147.
- [12] Nikolaos C Kyriakidis et al. “SARS-CoV-2 vaccines strategies: a comprehensive review of phase 3 candidates”. In: *npj Vaccines* 6.1 (2021), p. 28.
- [13] Hélène Kaplon et al. “Antibodies to watch in 2023”. In: *MAbs* 15.1 (2023), p. 2153410.
- [14] Ruei-Min Lu et al. “Development of therapeutic antibodies for the treatment of diseases”. In: *Journal of biomedical science* 27.1 (2020), pp. 1–30.
- [15] Melanie Senior. “Fresh from the biotech pipeline: fewer approvals, but biologics gain share”. In: *Nature Biotechnology* (2023), p. 1.

- [16] Laura L Hammitt et al. “Nirsevimab for prevention of RSV in healthy late-preterm and term infants”. In: *New England Journal of Medicine* 386.9 (2022), pp. 837–846.
- [17] Lawrence Steinman et al. “Ublituximab versus teriflunomide in relapsing multiple sclerosis”. In: *New England Journal of Medicine* 387.8 (2022), pp. 704–714.
- [18] Hussein A Tawbi et al. “Relatlimab and nivolumab versus nivolumab in untreated advanced melanoma”. In: *New England Journal of Medicine* 386.1 (2022), pp. 24–34.
- [19] Jeffrey V Ravetch and Jean-Pierre Kinet. “Fc receptors”. In: *Annual review of immunology* 9.1 (1991), pp. 457–492.
- [20] Alexandre Iannello and Ali Ahmad. “Role of antibody-dependent cell-mediated cytotoxicity in the efficacy of therapeutic anti-cancer monoclonal antibodies”. In: *Cancer and Metastasis Reviews* 24 (2005), pp. 487–499.
- [21] Ali Ahmad and Jose Menezes. “Antibody-dependent cellular cytotoxicity in HIV infections”. In: *The FASEB journal* 10.2 (1996), pp. 258–266.
- [22] Josiah T Ryman and Bernd Meibohm. “Pharmacokinetics of monoclonal antibodies”. In: *CPT: pharmacometrics & systems pharmacology* 6.9 (2017), pp. 576–588.
- [23] Yaghoub Safdari et al. “Antibody humanization methods—a review and update”. In: *Biotechnology and Genetic Engineering Reviews* 29.2 (2013), pp. 175–186.
- [24] Claire Marks et al. “Humanization of antibodies using a machine learning approach on large-scale repertoire data.” In: *Bioinformatics* 37.22 (2021), pp. 4041–4047.
- [25] Maryam Tabasinezhad et al. “Trends in therapeutic antibody affinity maturation: From in-vitro towards next-generation sequencing approaches”. In: *Immunology Letters* 212 (2019), pp. 106–113.
- [26] Jiabing Ma et al. “Bispecific antibodies: from research to clinical application”. In: *Frontiers in Immunology* (2021), p. 1555.
- [27] Nicolas Joubert et al. “Antibody–drug conjugates: the last decade”. In: *Pharmaceuticals* 13.9 (2020), p. 245.
- [28] Adam C Fisher et al. “The current scientific and regulatory landscape in advancing integrated continuous biopharmaceutical manufacturing”. In: *Trends in biotechnology* 37.3 (2019), pp. 253–267.
- [29] Dawn M Ecker, Susan Dana Jones, and Howard L Levine. “The therapeutic monoclonal antibody market”. In: *MAbs* 7.1 (2015), pp. 9–14.
- [30] Marie-Paule Lefranc et al. “IMGT unique numbering for immunoglobulin and T cell receptor variable domains and Ig superfamily V-like domains.” In: *Developmental and comparative immunology* 27.1 (2003), pp. 55–77.
- [31] Elvin A Kabat and Tai Te Wu. “Attempts to locate complementarity-determining residues in the variable positions of light and heavy chains”. In: *Annals of the New York Academy of Sciences* 190.1 (1971), pp. 382–393.
- [32] Pedro M Alzari, MB Lascombe, and RJ Poljak. “Three-dimensional structure of antibodies”. In: *Annual review of immunology* 6.1 (1988), pp. 555–580.

- [33] AG Amit et al. “Three-dimensional structure of an antigen-antibody complex at 2.8 Å resolution”. In: *Science* 233.4765 (1986), pp. 747–753.
- [34] Cristian Regep et al. “The H3 loop of antibodies shows unique structural characteristics”. In: *Proteins: Structure, Function, and Bioinformatics* 85.7 (2017), pp. 1311–1318.
- [35] Bora Guloglu and Charlotte M Deane. “Specific attributes of the VL domain influence both the structure and structural variability of CDR-H3 through steric effects”. In: *bioRxiv* (2023).
- [36] Alissa M Hummer, Brennan Abanades, and Charlotte M Deane. “Advances in computational structure-based antibody design”. In: *Current Opinion in Structural Biology* 74 (2022), p. 102379.
- [37] James Dunbar and Charlotte M Deane. “ANARCI: antigen receptor numbering and receptor classification”. In: *Bioinformatics* 32.2 (2016), pp. 298–300.
- [38] Tai Te Wu and Elvin A Kabat. “An analysis of the sequences of the variable regions of Bence Jones proteins and myeloma light chains and their implications for antibody complementarity”. In: *The Journal of experimental medicine* 132.2 (1970), pp. 211–250.
- [39] Cyrus Chothia et al. “Conformations of immunoglobulin hypervariable regions”. In: *Nature* 342.6252 (1989), pp. 877–883.
- [40] Véronique Giudicelli, Denys Chaume, and Marie-Paule Lefranc. “IMGT/GENE-DB: a comprehensive database for human and mouse immunoglobulin and T cell receptor genes”. In: *Nucleic acids research* 33.suppl_1 (2005), pp. D256–D261.
- [41] Bryan S Briney and James E Crowe Jr. “Secondary mechanisms of diversification in the human antibody repertoire”. In: *Frontiers in immunology* 4 (2013), p. 42.
- [42] Nilushi S De Silva and Ulf Klein. “Dynamics of B cells in germinal centres”. In: *Nature reviews immunology* 15.3 (2015), pp. 137–148.
- [43] Christopher J Jolly et al. “The targeting of somatic hypermutation”. In: *Seminars in immunology* 8.3 (1996), pp. 159–168.
- [44] Nai-Ying Zheng et al. “Intricate targeting of immunoglobulin somatic hypermutation maximizes the efficiency of affinity maturation”. In: *The Journal of experimental medicine* 201.9 (2005), pp. 1467–1478.
- [45] Louis A Clark et al. “Trends in antibody sequence changes during the somatic hypermutation process”. In: *The Journal of Immunology* 177.1 (2006), pp. 333–340.
- [46] Inbal Sela-Culang, Vered Kunik, and Yanay Ofran. “The structural basis of antibody-antigen recognition”. In: *Frontiers in immunology* 4 (2013), p. 302.
- [47] Cyrus Chothia et al. “Domain association in immunoglobulin molecules: the packing of variable domains”. In: *Journal of molecular biology* 186.3 (1985), pp. 651–663.
- [48] Cyrus Chothia and Arthur M Lesk. “Canonical structures for the hypervariable regions of immunoglobulins”. In: *Journal of molecular biology* 196.4 (1987), pp. 901–917.

- [49] Andrew CR Martin and Janet M Thornton. “Structural families in loops of homologous proteins: automatic classification, modelling and application to antibodies”. In: *Journal of molecular biology* 263.5 (1996), pp. 800–815.
- [50] Benjamin North, Andreas Lehmann, and Roland L Dunbrack Jr. “A new clustering of antibody CDR loop conformations”. In: *Journal of molecular biology* 406.2 (2011), pp. 228–256.
- [51] Simon Kelow et al. “A penultimate classification of canonical antibody CDR conformations”. In: *bioRxiv* (2022), pp. 2022–10.
- [52] Wing Ki Wong et al. “SCALOP: sequence-based antibody canonical loop structure annotation”. In: *Bioinformatics* 35.10 (2019), pp. 1774–1776.
- [53] Alexey Teplyakov et al. “Antibody modeling assessment II. Structures and models”. In: *Proteins: Structure, Function, and Bioinformatics* 82.8 (2014), pp. 1563–1582.
- [54] Jefferson Foote and Greg Winter. “Antibody framework residues affecting the conformation of the hypervariable loops”. In: *Journal of molecular biology* 224.2 (1992), pp. 487–499.
- [55] James Dunbar et al. “ABangle: characterising the VH-VL orientation in antibodies”. In: *Protein Engineering, Design & Selection* 26.10 (2013), pp. 611–620.
- [56] Enrique Vargas-Madrado and Enrique Paz-García. “An improved model of association for VH–VL immunoglobulin domains: asymmetries between VH and VL in the packing of some interface residues”. In: *Journal of Molecular Recognition* 16.3 (2003), pp. 113–120.
- [57] Alexander Bujotzek et al. “Prediction of VH–VL domain orientation for antibody variable domain modeling”. In: *Proteins: Structure, Function, and Bioinformatics* 83.4 (2015), pp. 681–695.
- [58] Claire Marks and Charlotte M Deane. “How repertoire data are changing antibody science.” In: *The Journal of biological chemistry* 295.29 (2020), pp. 9823–9837.
- [59] Scott D Boyd et al. “Measurement and clinical monitoring of human lymphocyte clonality by massively parallel VDJ pyrosequencing”. In: *Science translational medicine* 1.12 (2009), 12ra23–12ra23.
- [60] Simon Friedensohn, Tarik A Khan, and Sai T Reddy. “Advanced methodologies in high-throughput sequencing of immune repertoires”. In: *Trends in biotechnology* 35.3 (2017), pp. 203–214.
- [61] Michael A Quail et al. “A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers”. In: *BMC genomics* 13.1 (2012), pp. 1–13.
- [62] Cinque Soto et al. “High frequency of shared clonotypes in human B cell receptor repertoires”. In: *Nature* 566.7744 (2019), pp. 398–402.
- [63] Victor Greiff et al. “Learning the high-dimensional immunogenomic features that predict public and private antibody repertoires”. In: *The Journal of Immunology* 199.8 (2017), pp. 2985–2997.

- [64] Laura López-Santibáñez-Jácome, S Eréndira Avendaño-Vázquez, and Carlos Fabián Flores-Jasso. “The Pipeline Repertoire for Ig-Seq Analysis”. In: *Frontiers in Immunology* 10 (2019), p. 899.
- [65] Aleksandr Kovaltsuk et al. “Observed antibody space: a resource for data mining next-generation sequencing of antibody repertoires”. In: *The Journal of Immunology* 201.8 (2018), pp. 2502–2509.
- [66] Brian D Corrie et al. “iReceptor: A platform for querying and analyzing antibody/B-cell and T-cell receptor repertoire data across federated repositories.” In: *Immunological reviews* 284.1 (2018), pp. 24–41.
- [67] Tobias H Olsen, Iain H Moal, and Charlotte M Deane. “AbLang: an antibody language model for completing antibody sequences”. In: *Bioinformatics Advances* 2.1 (2022), vbac046.
- [68] Jeffrey A Ruffolo, Jeffrey J Gray, and Jeremias Sulam. “Deciphering antibody affinity maturation with language models and weakly supervised learning”. In: *arXiv preprint arXiv:2112.07782* (2021).
- [69] Jinwoo Leem et al. “Deciphering the language of antibodies using self-supervised learning”. In: *Patterns* 3.7 (2022), p. 100513.
- [70] Saikat Banerjee et al. “Evaluation of a novel multi-immunogen vaccine strategy for targeting 4E10/10E8 neutralizing epitopes on HIV-1 gp41 membrane proximal external region”. In: *Virology* 505 (2017), pp. 113–126.
- [71] Jackson S Turner et al. “SARS-CoV-2 mRNA vaccines induce persistent human germinal centre responses”. In: *Nature* 596.7870 (2021), pp. 109–113.
- [72] Cristina Parola, Daniel Neumeier, and Sai T Reddy. “Integrating high-throughput screening and sequencing for monoclonal antibody discovery and engineering”. In: *Immunology* 153.1 (2018), pp. 31–41.
- [73] Eve Richardson et al. “A computational method for immune repertoire mining that identifies novel binders from different clonotypes, demonstrated by identifying anti-pertussis toxoid antibodies.” In: *MAbs* 13.1 (2021), p. 1869406.
- [74] Brandon J DeKosky et al. “Large-scale sequence and structural comparisons of human naive and antigen-experienced antibody repertoires”. In: *Proceedings of the National Academy of Sciences* 113.19 (2016), E2636–E2645.
- [75] Matthew IJ Raybould et al. “Public Baseline and shared response structures support the theory of antibody repertoire functional commonality”. In: *PLoS computational biology* 17.3 (2021), e1008781.
- [76] Brandon J DeKosky et al. “High-throughput sequencing of the paired human immunoglobulin heavy and light chain repertoire”. In: *Nature biotechnology* 31.2 (2013), pp. 166–169.
- [77] Leonard D Goldstein et al. “Massively parallel single-cell B-cell receptor sequencing enables rapid discovery of diverse antigen-reactive antibodies”. In: *Communications Biology* 2.1 (2019), p. 304.
- [78] Michael Mor et al. “Multi-clonal SARS-CoV-2 neutralization by antibodies isolated from severe COVID-19 convalescent donors”. In: *PLoS Pathogens* 17.2 (2021), e1009165.

- [79] Hamish W King et al. “Single-cell analysis of human B cell maturation predicts how antibody class switching shapes selection dynamics”. In: *Science Immunology* 6.56 (2021), eabe6291.
- [80] David B Jaffe et al. “Functional antibodies exhibit light chain coherence”. In: *Nature* 611.7935 (2022), pp. 352–357.
- [81] Katie Kingwell. “T cell receptor therapeutics hit the immuno-oncology stage.” In: *Nature reviews. Drug Discovery* (2022).
- [82] Emily Y Yang and Khalid Shah. “Nanobodies: next generation of cancer diagnostics and therapeutics”. In: *Frontiers in Oncology* 10 (2020), p. 1182.
- [83] Gemma L Gordon et al. “A comparison of the binding sites of antibodies and single-domain antibodies”. In: *bioRxiv* (2023).
- [84] Wing Ki Wong, Jinwoo Leem, and Charlotte M Deane. “Comparative analysis of the CDR loops of antigen receptors”. In: *Frontiers in immunology* 10 (2019), p. 2454.
- [85] Laura S Mitchell and Lucy J Colwell. “Comparative analysis of nanobody sequence and structure data”. In: *Proteins: Structure, Function, and Bioinformatics* 86.7 (2018), pp. 697–706.
- [86] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
- [87] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *Nature* 521.7553 (2015), pp. 436–444.
- [88] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [89] Robert Hecht-Nielsen. “Theory of the backpropagation neural network”. In: *Neural networks for perception*. Elsevier, 1992, pp. 65–93.
- [90] Honglak Lee et al. “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations”. In: *Proceedings of the 26th annual international conference on machine learning* (2009), pp. 609–616.
- [91] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017), pp. 84–90.
- [92] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5 (1989), pp. 359–366.
- [93] George Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.
- [94] Anders Krogh and John Hertz. “A simple weight decay can improve generalization”. In: *Advances in neural information processing systems* 4 (1991).
- [95] Yann LeCun et al. “Handwritten digit recognition with a back-propagation network”. In: *Advances in neural information processing systems* 2 (1989).
- [96] Marco Gori, Gabriele Monfardini, and Franco Scarselli. “A new model for learning in graph domains”. In: *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* 2 (2005), pp. 729–734.

- [97] Jie Zhou et al. “Graph neural networks: A review of methods and applications”. In: *AI open* 1 (2020), pp. 57–81.
- [98] Michael M Bronstein et al. “Geometric deep learning: going beyond euclidean data”. In: *IEEE Signal Processing Magazine* 34.4 (2017), pp. 18–42.
- [99] Nathaniel Thomas et al. “Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds”. In: *arXiv preprint arXiv:1802.08219* (2018).
- [100] Mario Geiger and Tess Smidt. “e3nn: Euclidean neural networks”. In: *arXiv preprint arXiv:2207.09453* (2022).
- [101] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).
- [102] Fabian Fuchs et al. “Se (3)-transformers: 3d roto-translation equivariant attention networks”. In: *Advances in neural information processing systems* 33 (2020), pp. 1970–1981.
- [103] Marc Finzi, Max Welling, and Andrew Gordon Wilson. “A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups”. In: *International conference on machine learning* (2021), pp. 3318–3328.
- [104] Nadav Dym and Haggai Maron. “On the universality of rotation equivariant point cloud networks”. In: *arXiv preprint arXiv:2010.02449* (2020).
- [105] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. “E (n) equivariant graph neural networks”. In: *International conference on machine learning* (2021), pp. 9323–9332.
- [106] Congyue Deng et al. “Vector neurons: A general framework for so (3)-equivariant networks”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 12200–12209.
- [107] Bowen Jing et al. “Learning from protein structure with geometric vector perceptrons”. In: *arXiv preprint arXiv:2009.01411* (2020).
- [108] Chloe Hsu et al. “Learning inverse folding from millions of predicted structures”. In: *International Conference on Machine Learning* (2022), pp. 8946–8970.
- [109] Bruce Alberts. *Molecular biology of the cell*. Garland science, 2017.
- [110] Lukasz Slabinski et al. “The challenge of protein structure determination—lessons from structural genomics”. In: *Protein Science* 16.11 (2007), pp. 2472–2482.
- [111] Brian Kuhlman and Philip Bradley. “Advances in protein structure prediction and design”. In: *Nature Reviews Molecular Cell Biology* 20.11 (2019), pp. 681–697.
- [112] Claudio N Cavasotto and Sharangdhar S Phatak. “Homology modeling in drug discovery: current trends and applications”. In: *Drug discovery today* 14.13-14 (2009), pp. 676–683.
- [113] Wynne J Browne et al. “A possible three-dimensional structure of bovine α -lactalbumin based on that of hen’s egg-white lysozyme”. In: *Journal of molecular biology* 42.1 (1969), pp. 65–86.
- [114] Manuel C Peitsch. “Protein modeling by E-mail”. In: *Bio/technology* 13.7 (1995), pp. 658–660.

- [115] Torsten Schwede et al. “SWISS-MODEL: an automated protein homology-modeling server”. In: *Nucleic acids research* 31.13 (2003), pp. 3381–3385.
- [116] Andrew Waterhouse et al. “SWISS-MODEL: homology modelling of protein structures and complexes”. In: *Nucleic acids research* 46.W1 (2018), W296–W303.
- [117] John Moult et al. “Critical assessment of methods of protein structure prediction (CASP)—round X”. In: *Proteins: Structure, Function, and Bioinformatics* 82 (2014), pp. 1–6.
- [118] Jinwoo Leem et al. “ABodyBuilder: Automated antibody structure prediction with data-driven accuracy estimation”. In: *MAbs* 8.7 (2016), pp. 1259–1268.
- [119] Bernard R Brooks et al. “CHARMM: the biomolecular simulation program”. In: *Journal of computational chemistry* 30.10 (2009), pp. 1545–1614.
- [120] Romelia Salomon-Ferrer, David A Case, and Ross C Walker. “An overview of the Amber biomolecular simulation package”. In: *Wiley Interdisciplinary Reviews: Computational Molecular Science* 3.2 (2013), pp. 198–210.
- [121] Jianyi Yang et al. “The I-TASSER Suite: protein structure and function prediction”. In: *Nature methods* 12.1 (2015), pp. 7–8.
- [122] Julia Koehler Leman et al. “Macromolecular modeling and design in Rosetta: recent methods and frameworks”. In: *Nature methods* 17.7 (2020), pp. 665–680.
- [123] Saulo de Oliveira and Charlotte Deane. “Co-evolution techniques are reshaping the way we do structural bioinformatics”. In: *F1000Research* 6 (2017).
- [124] Charles Yanofsky, Virginia Horn, and Deanna Thorpe. “Protein structure relationships revealed by mutational analysis”. In: *Science* 146.3651 (1964), pp. 1593–1594.
- [125] Faruck Morcos et al. “Direct-coupling analysis of residue coevolution captures native contacts across many protein families”. In: *Proceedings of the National Academy of Sciences* 108.49 (2011), E1293–E1301.
- [126] Debora S Marks et al. “Protein 3D structure computed from evolutionary sequence variation”. In: *PloS one* 6.12 (2011), e28766.
- [127] David T Jones et al. “MetaPSICOV: combining coevolution methods for accurate prediction of contacts and long range hydrogen bonding in proteins”. In: *Bioinformatics* 31.7 (2015), pp. 999–1006.
- [128] Sheng Wang et al. “Accurate de novo prediction of protein contact map by ultra-deep learning model”. In: *PLoS computational biology* 13.1 (2017), e1005324.
- [129] Jinbo Xu. “Distance-based protein folding powered by deep learning”. In: *Proceedings of the National Academy of Sciences* 116.34 (2019), pp. 16856–16865.
- [130] Andrew W Senior et al. “Improved protein structure prediction using potentials from deep learning”. In: *Nature* 577.7792 (2020), pp. 706–710.
- [131] Jianyi Yang et al. “Improved protein structure prediction using predicted interresidue orientations”. In: *Proceedings of the National Academy of Sciences* 117.3 (2020), pp. 1496–1503.

- [132] Andriy Kryshchuk et al. “Critical assessment of methods of protein structure prediction (CASP)—Round XIII”. In: *Proteins: Structure, Function, and Bioinformatics* 87.12 (2019), pp. 1011–1020.
- [133] Arne Elofsson. “Progress at protein structure prediction, as seen in CASP15”. In: *Current Opinion in Structural Biology* 80 (2023), p. 102594.
- [134] Ewen Callaway. “‘It will change everything’: DeepMind’s AI makes gigantic leap in solving protein structures”. In: *Nature* 588.7837 (2020), pp. 203–205.
- [135] John Ingraham et al. “Learning protein structure with a differentiable simulator”. In: *International Conference on Learning Representations* (2019).
- [136] Mohammed AlQuraishi. “End-to-end differentiable learning of protein structure”. In: *Cell systems* 8.4 (2019), pp. 292–301.
- [137] Roshan M Rao et al. “MSA transformer”. In: *International Conference on Machine Learning* (2021), pp. 8844–8856.
- [138] Viktor Hornak et al. “Comparison of multiple Amber force fields and development of improved protein backbone parameters”. In: *Proteins: Structure, Function, and Bioinformatics* 65.3 (2006), pp. 712–725.
- [139] Minkyung Baek et al. “Accurate prediction of protein structures and interactions using a three-track neural network”. In: *Science* 373.6557 (2021), pp. 871–876.
- [140] Zeming Lin et al. “Evolutionary-scale prediction of atomic-level protein structure with a language model”. In: *Science* 379.6637 (2023), pp. 1123–1130.
- [141] Ruidong Wu et al. “High-resolution de novo structure prediction from primary sequence”. In: *bioRxiv* (2022).
- [142] Xiaomin Fang et al. “Helixfold-single: Msa-free protein structure prediction by using protein language model as an alternative”. In: *arXiv preprint arXiv:2207.13921* (2022).
- [143] Shenggan Cheng et al. “Fastfold: Reducing alphafold training time from 11 days to 67 hours”. In: *arXiv preprint arXiv:2203.00854* (2022).
- [144] Ziyao Li et al. “Uni-Fold: an open-source platform for developing protein folding models beyond AlphaFold”. In: *bioRxiv* (2022).
- [145] Gustaf Ahdritz et al. “OpenFold: Retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization”. In: *bioRxiv* (2022).
- [146] Richard Evans et al. “Protein complex prediction with AlphaFold-Multimer”. In: *bioRxiv* (2021).
- [147] Usman Ghani et al. “Improved docking of protein models by a combination of alphafold2 and cluspro”. In: *bioRxiv* (2021).
- [148] L Steven Johnson, Sean R Eddy, and Elon Portugaly. “Hidden Markov model speed heuristic and iterative HMM search procedure”. In: *BMC bioinformatics* 11 (2010), pp. 1–8.
- [149] Michael Remmert et al. “HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment”. In: *Nature methods* 9.2 (2012), pp. 173–175.
- [150] Martin Steinegger and Johannes Söding. “MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets”. In: *Nature biotechnology* 35.11 (2017), pp. 1026–1028.

- [151] Milot Mirdita et al. “ColabFold: making protein folding accessible to all”. In: *Nature Methods* (2022), pp. 1–4.
- [152] Jae Hyeon Lee et al. “EquiFold: Protein Structure Prediction with a Novel Coarse-Grained Structure Representation”. In: *bioRxiv* (2022).
- [153] Minkyung Baek et al. “Efficient and accurate prediction of protein structure using RoseTTAFold2”. In: *bioRxiv* (2023).
- [154] James Dunbar et al. “SAbDab: the structural antibody database”. In: *Nucleic acids research* 42.D1 (2014), pp. D1140–D1146.
- [155] Jinwoo Leem et al. “STCRDab: the structural T-cell receptor database”. In: *Nucleic acids research* 46.D1 (2018), pp. D406–D412.
- [156] Constantin Schneider, Matthew IJ Raybould, and Charlotte M Deane. “SAbDab in the age of biotherapeutics: updates including SAbDab-nano, the nanobody structure tracker”. In: *Nucleic acids research* 50.D1 (2022), pp. D1368–D1372.
- [157] Alexander Jarasch et al. “Developability assessment during the selection of novel therapeutic antibodies”. In: *Journal of pharmaceutical sciences* 104.6 (2015), pp. 1885–1898.
- [158] Matthew I J Raybould et al. “Five computational developability guidelines for therapeutic antibody profiling”. In: *Proceedings of the National Academy of Sciences* 116.10 (2019), pp. 4025–4030.
- [159] Adriana-Michelle Wolf Pérez et al. “Assessment of therapeutic antibody developability by combinations of in vitro and in silico methods”. In: *Therapeutic Antibodies: Methods and Protocols* (2022), pp. 57–113.
- [160] Sarah A Robinson et al. “Epitope profiling using computational structural modelling demonstrated on coronavirus-binding antibodies”. In: *PLoS computational biology* 17.12 (2021), e1009675.
- [161] Fabian C Spoendlin et al. “Improved computational epitope profiling using structural models identifies a broader diversity of antibodies that bind the same epitope”. In: *bioRxiv* (2023), pp. 2023–06.
- [162] Aleksandr Kovaltsuk et al. “Structural diversity of B-cell receptor repertoires along the B-cell differentiation axis in humans and mice”. In: *PLoS computational biology* 16.2 (2020), e1007636.
- [163] Wiktoria Wilman et al. “Machine-designed biotherapeutics: opportunities, feasibility and advantages of deep learning in computational antibody discovery”. In: *Briefings in Bioinformatics* 23.4 (2022), bbac267.
- [164] Amir Shanehsazzadeh et al. “Unlocking de novo antibody design with generative artificial intelligence”. In: *bioRxiv* (2023).
- [165] Tileli Amimeur et al. “Designing feature-controlled humanoid antibody discovery libraries using generative adversarial networks”. In: *bioRxiv* (2020).
- [166] Richard W Shuai, Jeffrey A Ruffolo, and Jeffrey J Gray. “Generative language modeling for antibody design”. In: *bioRxiv* (2021).
- [167] Brian L Hie et al. “Efficient evolution of human antibodies from general protein language models”. In: *Nature Biotechnology* (2023).

- [168] Justas Dauparas et al. “Robust deep learning-based protein sequence design using ProteinMPNN”. In: *Science* 378.6615 (2022), pp. 49–56.
- [169] Srivamshi Pittala and Chris Bailey-Kellogg. “Learning context-aware structural representations to predict antigen and antibody binding interfaces”. In: *Bioinformatics* 36.13 (2020), pp. 3996–4003.
- [170] Lewis Chinery et al. “Paragraph—antibody paratope prediction using graph neural networks with minimal feature vectors”. In: *Bioinformatics* 39.1 (2023), btac732.
- [171] Wing Ki Wong et al. “Ab-Ligity: identifying sequence-dissimilar antibodies that bind to the same epitope”. In: *MAbs* 13.1 (2021), p. 1873478.
- [172] Yoonjoo Choi and Charlotte M Deane. “FREAD revisited: accurate loop structure prediction using a database search algorithm”. In: *Proteins: Structure, Function, and Bioinformatics* 78.6 (2010), pp. 1431–1440.
- [173] Benjamin Webb and Andrej Sali. “Comparative protein structure modeling using MODELLER”. In: *Current protocols in bioinformatics* 54.1 (2016), pp. 5–6.
- [174] Claire Marks et al. “Sphinx: merging knowledge-based and ab initio approaches to improve protein loop prediction”. In: *Bioinformatics* 33.9 (2017), pp. 1346–1353.
- [175] Jinwoo Leem et al. “Antibody side chain conformations are position-dependent”. In: *Proteins: Structure, Function, and Bioinformatics* 86.4 (2018), pp. 383–392.
- [176] Jeffrey A Ruffolo et al. “Geometric potentials from deep learning improve prediction of CDR H3 loop structures”. In: *Bioinformatics* 36.Supplement_1 (2020), pp. i268–i275.
- [177] Hahnbeom Park et al. “Simultaneous optimization of biomolecular energy functions on features from small molecules and macromolecules”. In: *Journal of chemical theory and computation* 12.12 (2016), pp. 6201–6212.
- [178] Jeffrey A. Ruffolo, Jeremias Sulam, and Jeffrey J. Gray. “Antibody structure prediction using interpretable deep learning”. In: *Patterns* (2021), p. 100406.
- [179] Jeffrey A Ruffolo and Jeffrey J Gray. “Fast, accurate antibody structure prediction from deep learning on massive set of natural antibodies”. In: *Biophysical Journal* 121.3 (2022), 155a–156a.
- [180] Yining Wang et al. “xTrimABFold: Improving Antibody Structure Prediction without Multiple Sequence Alignments”. In: *arXiv preprint arXiv:2212.00735* (2022).
- [181] Jiaxiang Wu et al. “tFold-Ab: Fast and Accurate Antibody Structure Prediction without Sequence Homologs”. In: *bioRxiv* (2022).
- [182] Chao Peng et al. “AbFold—an AlphaFold Based Transfer Learning Model for Accurate Antibody Structure Prediction”. In: *bioRxiv* (2023).
- [183] Matthew IJ Raybould, Wing Ki Wong, and Charlotte M Deane. “Antibody–antigen complex modelling in the era of immunoglobulin repertoire sequencing”. In: *Molecular Systems Design & Engineering* 4.4 (2019), pp. 679–688.
- [184] Brian G Pierce et al. “ZDOCK server: interactive docking prediction of protein–protein complexes and symmetric multimers”. In: *Bioinformatics* 30.12 (2014), pp. 1771–1773.

- [185] Konrad Krawczyk et al. “Antibody i-Patch prediction of the antibody binding site improves rigid local antibody–antigen docking”. In: *Protein Engineering, Design & Selection* 26.10 (2013), pp. 621–629.
- [186] Francesco Ambrosetti et al. “Modeling antibody-antigen complexes by information-driven docking”. In: *Structure* 28.1 (2020), pp. 119–129.
- [187] Ana Davila et al. “AbAdapt: an adaptive approach to predicting antibody–antigen complex structures from sequence”. In: *Bioinformatics Advances* 2.1 (2022), vbac015.
- [188] Magnus Haraldson Hoie et al. “DiscoTope-3.0-Improved B-cell epitope prediction using AlphaFold2 modeling and inverse folding latent representations”. In: *bioRxiv* (2023).
- [189] Constantin Schneider et al. “DLAB: deep learning methods for structure-based virtual screening of antibodies”. In: *Bioinformatics* 38.2 (2022), pp. 377–383.
- [190] Philippe A Robert et al. “Unconstrained generation of synthetic antibody–antigen structures to guide machine learning methodology for antibody specificity prediction”. In: *Nature Computational Science* 2.12 (2022), pp. 845–865.
- [191] Alissa M Hummer et al. “Investigating the Volume and Diversity of Data Needed for Generalizable Antibody-Antigen $\Delta\Delta G$ Prediction”. In: *bioRxiv* (2023).
- [192] Mark L Chiu et al. “Antibody Structure and Function: The Basis for Engineering Therapeutics”. In: *Antibodies* 8.4 (2019), p. 55.
- [193] Helen M Berman et al. “The protein data bank”. In: *Nucleic acids research* 28.1 (2000), pp. 235–242.
- [194] Liyuan Liu et al. “On the Variance of the Adaptive Learning Rate and Beyond”. In: *International Conference on Learning Representations* (2020).
- [195] James A Maier et al. “ff14SB: improving the accuracy of protein side chain and backbone parameters from ff99SB”. In: *Journal of chemical theory and computation* 11.8 (2015), pp. 3696–3713.
- [196] Peter Eastman et al. “OpenMM 7: Rapid development of high performance algorithms for molecular dynamics”. In: *PLoS computational biology* 13.7 (2017), e1005659.
- [197] Monica L Fernández-Quintero et al. “Challenges in antibody structure prediction”. In: *MAbs* 15.1 (2023), p. 2175319.
- [198] Wing Ki Wong et al. “TCRBuilder: multi-state T-cell receptor structure prediction”. In: *Bioinformatics* 36.11 (2020), pp. 3580–3581.
- [199] Tomer Cohen, Matan Halfon, and Dina Schneidman-Duhovny. “NanoNet: Rapid and accurate end-to-end nanobody modeling by deep learning”. In: *Frontiers in immunology* 13 (2022), p. 958584.
- [200] Wolfgang Kabsch. “A solution for the best rotation to relate two sets of vectors”. In: *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), pp. 922–923.
- [201] Eduard Schreiner et al. “Stereochemical errors and their implications for molecular dynamics simulations”. In: *BMC bioinformatics* 12.1 (2011), pp. 1–9.

- [202] Dimitri Schritt et al. “Repertoire Builder: high-throughput structural modeling of B and T cell receptors”. In: *Molecular Systems Design & Engineering* 4.4 (2019), pp. 761–768.
- [203] Johannes KX Maier and Paul Labute. “Assessment of fully automated antibody homology modeling protocols in molecular operating environment”. In: *Proteins: Structure, Function, and Bioinformatics* 82.8 (2014), pp. 1599–1610.
- [204] Eran Eyal et al. “The limit of accuracy of protein modeling: influence of crystal packing on protein structure”. In: *Journal of molecular biology* 351.2 (2005), pp. 431–442.
- [205] Matthew Z Tien et al. “Maximum allowed solvent accessibilities of residues in proteins”. In: *PloS one* 8.11 (2013), e80635.
- [206] Shira Warszawski et al. “Optimizing antibody affinity and stability by the automated design of the variable light-heavy chain interfaces”. In: *PLOS Computational Biology* 15.8 (2019), pp. 1–24.
- [207] Shabdita Vatsa. “In silico prediction of post-translational modifications in therapeutic antibodies”. In: *MAbs* 14.1 (2022), p. 2023938.
- [208] Rob van der Kant et al. “Prediction and Reduction of the Aggregation of Monoclonal Antibodies.” In: *Journal of molecular biology* 429.8 (2017), pp. 1244–1261.
- [209] Bo Wang et al. “Facile Discovery of a Diverse Panel of Anti-Ebola Virus Antibodies by Immune Repertoire Mining”. In: *Scientific Reports* 5.1 (2015), p. 13926.
- [210] Xiaolong Tian et al. “Deep Mining of Human Antibody Repertoires: Concepts, Methodologies, and Applications”. In: *Small Methods* 4.10 (2020), p. 2000451.
- [211] Yi-Chun Hsiao et al. “Immune repertoire mining for rapid affinity optimization of mouse monoclonal antibodies.” In: *MAbs* 11.4 (2019), pp. 735–746.
- [212] Konrad Krawczyk et al. “Looking for therapeutic antibodies in next-generation sequencing repositories”. In: *MAbs* 11.7 (2019), pp. 1197–1205.
- [213] Konrad Krawczyk et al. “Structurally Mapping Antibody Repertoires.” In: *Frontiers in immunology* 9 (2018), p. 1698.
- [214] Michel van Kempen et al. “Fast and accurate protein structure search with Foldseek”. In: *Nature Biotechnology* (2023), pp. 1–4.
- [215] Xinyang Li et al. “Comparative Analysis of Immune Repertoires between Bactrian Camel’s Conventional and Heavy-Chain Antibodies”. In: *PloS one* 11.9 (2016), e0161801.
- [216] Martin M Corcoran et al. “Production of individualized V gene databases reveals high levels of immunoglobulin genetic diversity”. In: *Nature Communications* 7.1 (2016), p. 13642.
- [217] Ang Cui et al. “A model of somatic hypermutation targeting in mice based on high-throughput Ig sequencing data”. In: *The Journal of Immunology* 197.9 (2016), pp. 3566–3574.

- [218] Erik L Johnson et al. “Sequencing HIV-neutralizing antibody exons and introns reveals detailed aspects of lineage maturation”. In: *Nature Communications* 9.1 (2018), p. 4136.
- [219] Joana P Bernardes et al. “Longitudinal Multi-omics Analyses Identify Responses of Megakaryocytes, Erythroid Cells, and Plasmablasts as Hallmarks of Severe COVID-19”. In: *Immunity* 53.6 (2020), pp. 1296–1314.
- [220] Marie-Paule Lefranc. “Unique database numberings system for immunogenetic analysis”. In: *Immunology Today* 18.11 (1997), p. 509.
- [221] Christiam Camacho et al. “BLAST+: architecture and applications”. In: *BMC Bioinformatics* 10.1 (2009), p. 421.
- [222] Limin Fu et al. “CD-HIT: accelerated for clustering the next-generation sequencing data”. In: *Bioinformatics* 28.23 (2012), pp. 3150–3152.
- [223] Martin Steinegger and Johannes Söding. “Clustering huge protein sequence sets in linear time”. In: *Nature Communications* 9.1 (2018), p. 2542.
- [224] S Henikoff and J G Henikoff. “Amino acid substitution matrices from protein blocks.” In: *Proceedings of the National Academy of Sciences of the United States of America* 89.22 (1992), pp. 10915–10919.
- [225] Weizhong Li, Lukasz Jaroszewski, and Adam Godzik. “Clustering of highly homologous sequences to reduce the size of large protein databases ”. In: *Bioinformatics* 17.3 (2001), pp. 282–283.
- [226] T F Smith and M S Waterman. “Identification of common molecular subsequences”. In: *Journal of Molecular Biology* 147.1 (1981), pp. 195–197.
- [227] Jakub Młokosiewicz et al. “AbDiver: a tool to explore the natural antibody landscape to aid therapeutic design”. In: *Bioinformatics* 38.9 (2022), pp. 2628–2630.
- [228] Torbjørn Rognes et al. “CompAIRR: ultra-fast comparison of adaptive immune receptor repertoires by exact and approximate sequence matching”. In: *Bioinformatics* 38.17 (2022), pp. 4230–4232.
- [229] Matthew I J Raybould et al. “Thera-SAbDab: the Therapeutic Structural Antibody Database”. In: *Nucleic Acids Research* 48.D1 (2020), pp. D383–D388.
- [230] Roy Frostig, Matthew James Johnson, and Chris Leary. “Compiling machine learning programs via high-level tracing”. In: *Systems for Machine Learning* 4.9 (2018).
- [231] Florian Rubelt et al. “Adaptive Immune Receptor Repertoire Community recommendations for sharing immune-repertoire sequencing data”. In: *Nature immunology* 18.12 (2017), pp. 1274–1278.
- [232] Wanwisa Dejnirattisai et al. “The antigenic anatomy of SARS-CoV-2 receptor binding domain.” In: *Cell* 184.8 (2021), pp. 2183–2200.
- [233] Andrea P Espejo et al. “Review of current advances in serologic testing for COVID-19”. In: *Am J Clin Pathol* 154.3 (2020), pp. 293–304.
- [234] Andrew J Pollard and Else M Bijker. “A guide to vaccinology: from basic principles to new developments”. In: *Nat Rev Immunol* 21.2 (2021), pp. 83–100.

- [235] Matthew IJ Raybould et al. “CoV-AbDab: the coronavirus antibody database”. In: *Bioinformatics* 37.5 (2021), pp. 734–735.
- [236] Konrad Krawczyk, Andrew Buchanan, and Paolo Marcatili. “Data mining patented antibody sequences”. In: *MAbs* 13.1 (2021), p. 1892366.
- [237] Ward Fleri et al. “The immune epitope database and analysis resource in epitope discovery and synthetic vaccine design”. In: *Front Immunol* 8 (2017), p. 278.
- [238] Eric W Sayers et al. “Database resources of the national center for biotechnology information”. In: *Nucleic Acids Res* 50.D1 (2021), pp. D20–D26.
- [239] Peter JA Cock et al. “Biopython: freely available Python tools for computational molecular biology and bioinformatics”. In: *Bioinformatics* 25.11 (2009), pp. 1422–1423.
- [240] Morgan SA Gilman et al. “Rapid profiling of RSV antibody repertoires from the memory B cells of naturally infected adult donors”. In: *Sci Immunol* 1.6 (2016), eaaj1879.
- [241] Warren L DeLano et al. “Pymol: An open-source molecular graphics tool”. In: *CCP4 Newsl. Protein Crystallogr* 40.1 (2002), pp. 82–92.
- [242] Harry C Jubb et al. “Arpeggio: a web server for calculating and visualising interatomic interactions in protein structures”. In: *J Mol Biol* 429.3 (2017), pp. 365–371.
- [243] Maryam Mukhamedova et al. “Vaccination with prefusion-stabilized respiratory syncytial virus fusion protein induces genetically and antigenically diverse antibody responses”. In: *Immunity* 54.4 (2021), pp. 769–780.
- [244] Ellen L Shrock et al. “Germline-encoded amino acid-binding motifs drive immunodominant public antibody responses”. In: *Science* 380.6640 (2023), eadc9498.
- [245] R OpenAI. “GPT-4 technical report”. In: *arXiv* (2023), pp. 2303–08774.

Appendices



Machine learning datasets

A.1 ABlooper data

A.1.1 Rosetta Antibody Benchmark

1dlfH	1fnsH	1gigH	1jfqH	1jptH	1mfaH	1mlbB	1mqkH
1nlbH	1oaqH	1seqH	1x9qA	2adfH	2d7tH	2e27H	2fb4H
2fbjH	2r8sH	2v17H	2vxvH	2w60A	2xwtA	2ypvH	3e8uH
3eo9H	3g5yB	3gizH	3gnmH	3go1H	3hc4H	3hntH	3i9gH
3iflH	3lizH	3lmjH	3m8oH	3mlrH	3mxwH	3npsB	3oz9H
3p0yH	3t65B	3umtA	3v0wH	4f57H	4h0hB	4h20H	4hpyH
4nzuH							

Table A.1: Rosetta Antibody Benchmark (RAB) test set from Chapter 2. The first four characters form the PDB ID, the final character is the heavy chain ID.

A.1.2 SAbDab Latest Structures

5zmjH	6lfxA	6lunA	6lunC	6o1fH	6o2aH	6o2aC	6o2aO
6o2aA	6o2bH	6o2bA	6o2bC	6o2bO	6rloA	6rloG	6rloC
6rloE	6srxH	6srxI	6ss5H	6uy3A	6v4qH	6vc9H	6vryH
6vunA	6vy4C	6vy4H	6w4wH	6w5dA	6w5dH	6wk4A	6wklA
6wkmH	6wrpH	6wzlA	6wzlC	6wzmA	6wzmC	6x1sH	6x1sC
6x1sE	6x1sA	6x7wH	6x7wA	6x8sH	6x8uH	6y6cC	6y6cD
6yioH	6yxkA	6z7xB	6z7yB	6z7yD	6zvfH	7ah1A	7b3oH
7bxvH	7c88A	7c88H	7ce2Z	7cjfA	7dc7A	7deuB	7dm1F
7dm1D	7dunH	7e8mH	7eamH	7eamC	7jlkA	7jlkH	7jumJ
7jumH	7jumI	7jwqC	7jwqA	7k8nA	7ke1C	7ke1A	7kfvF
7kfvH	7kfvC	7km6h	7km6H	7kmgD	7kmgA	7kmhA	7kmiA

7kpgH	7kq7H	7kylX	7kylH	7lcvA	7lkaC	7lkaH	7lkaE
7lkaA	7lm8H	7lm9H	7lopW	7lsfH	7lsgH	7lxyM	7lxyE
7lxyF	7m51H	7m52H	7m53H	7m55H	7mf1H	7negH	7nehH
7nx6H	7nx8H						

Table A.2: SAbDab Latest Structures (SLS) test set from Chapter 2. The first four characters form the PDB ID, the final character is the heavy chain ID.

A.1.3 Train and validation

12e8H	12e8P	15c8H	1a14H	1a2yB	1a3lH	1a3rH	1a4jB
1a4jH	1a4kB	1a4kH	1a6tB	1a6tD	1a6uH	1a6vI	1a6vJ
1a6vH	1a6wH	1a7nH	1a7oH	1a7pH	1a7qH	1a7rH	1ad0D
1ad0B	1ad9B	1ad9H	1ai1H	1aifB	1aifH	1aj7H	1ap2D
1ap2B	1aqkH	1axsH	1axsB	1ay1H	1b2wH	1b4jH	1bafH
1bbdH	1bfoB	1bfoH	1bfoF	1bfoD	1bfvH	1bgxH	1bj1H
1bj1K	1blnD	1blnB	1bogB	1bqlH	1bvkB	1bvkE	1bvlA
1bvlC	1bz7B	1c08B	1c12B	1c1eH	1c5bH	1c5cH	1c5dB
1c5dH	1cbvH	1ce1H	1cf8H	1cfnB	1cfqB	1cfvH	1cgsH
1cicD	1cicB	1ck0H	1cloH	1clyH	1clzH	1cr9H	1ct8D
1ct8B	1cu4H	1cz8Y	1d5bB	1d5bH	1d5iH	1d6vH	1dbaH
1dbbH	1dbjH	1dbmH	1deeD	1deeF	1deeB	1dfbH	1dn0D
1dqjB	1dqlH	1dqmH	1dqqB	1dqqD	1dsfH	1dvfB	1dvfD
1e4wH	1e4xH	1e4xI	1e6oH	1eapB	1egjH	1ehlH	1ejoH
1emtH	1eo8H	1etzH	1etzB	1ezvX	1f3dK	1f3dH	1f4wH
1f4xH	1f4yH	1f58H	1f8tH	1f90H	1faiH	1fdlH	1fe8H
1fe8I	1fe8J	1fgnH	1fj1D	1fj1B	1fl3A	1fl3H	1fl5H
1fl5B	1fl6H	1fl6B	1flrH	1fn4B	1fn4D	1forH	1frgH
1fskI	1fskL	1fskF	1fskC	1fvcD	1fvcB	1fvdB	1fvdD
1fveB	1fveD	1g7hB	1g7iB	1g7jB	1g7lB	1g7mB	1g9mH
1g9nH	1gafH	1gc1H	1ggbH	1ggcH	1ggiJ	1ggiH	1ghfH
1gpoI	1gpoH	1h0dB	1h3pH	1hezB	1hezD	1hh6B	1hh9B
1hi6B	1hilD	1hilB	1himL	1himM	1hklH	1hq4D	1hq4B
1hzhH	1hzhK	1i7zB	1i7zD	1i8iB	1i8kB	1i8mH	1i8mB
1i9iH	1i9jH	1iaiH	1iaiI	1ibgH	1ic4H	1ic5H	1ic7H
1ifhH	1igcH	1igfJ	1igfH	1igiH	1igjD	1igjB	1igmH
1igtD	1igtB	1ikfH	1il1A	1indH	1ineH	1iqdB	1iqwH
1it9H	1j05B	1j05H	1j1oH	1j1pH	1j1xH	1jguH	1jgvH
1jhlH	1jn6B	1jnhF	1jnhH	1jnhD	1jnhB	1jp5A	1jp5B
1jrhH	1jv5B	1k4cA	1k4dA	1k6qH	1kb5H	1kb9J	1kc5H
1kcrH	1kcsH	1kcuH	1kcvH	1kegH	1kelH	1kemH	1kfaH
1kiqB	1kirB	1kn2H	1kn4H	1ktrL	1kyoU	1kyoJ	1l7iH
1l7tH	1lk3I	1lk3H	1lmkG	1lmkE	1lmkC	1lmkA	1lo0Y
1lo0H	1lo2Y	1lo2H	1lo3H	1lo3Y	1m71B	1m7iB	1mamH
1mcpH	1mexH	1mfbH	1mfcH	1mfdH	1mh5H	1mh5B	1mhpX

1mhpH	1mimH	1mj8H	1mjjB	1mjjH	1mjuH	1mnuH	1moeB
1moeA	1mpaH	1mrdH	1mreH	1mrfH	1mvuB	1n0xK	1n0xH
1n4xI	1n4xH	1n7mL	1n8zB	1nakI	1nakH	1nbvH	1nbyB
1nbzB	1nc2B	1nc2D	1nc4B	1nc4D	1ncaH	1ncbH	1nccH
1ncdH	1ncwH	1nd0F	1nd0D	1nd0B	1nd0H	1ndgB	1ndmB
1nfdF	1nfdH	1ngpH	1ngqH	1ngwB	1ngwH	1ngxH	1ngxB
1ngyB	1ngzB	1nj9B	1nj9H	1nl0H	1nldH	1nmbH	1nmcB
1nmcH	1nsnH	1om3H	1om3K	1op3M	1op3H	1opgH	1orsB
1ospH	1otsC	1otsE	1p2cB	1p2cE	1p4bH	1p4iH	1p7kB
1p84J	1pg7Z	1pg7X	1plgH	1pskH	1pz5B	1q0xH	1q0yH
1q1jI	1q1jH	1q72H	1q9oB	1q9oD	1q9wD	1q9wB	1qblH
1qbmH	1qfuH	1qkzH	1qokA	1qygH	1r0aH	1r3iH	1r3jB
1r3kB	1r3lB	1rfdH	1riuH	1rivH	1rjlB	1rmfH	1rukH
1rumH	1rupH	1ruqH	1rurH	1rz7H	1rz8D	1rz8B	1rzjH
1rzKH	1s3kH	1s5hB	1s5iH	1sbsH	1sm3H	1svzB	1svzA
1sy6H	1t3fB	1t4kD	1t4kB	1t66D	1t66H	1tetH	1tpxB
1tqbB	1tqcB	1tztI	1tztH	1tzhH	1tzhB	1tziB	1u6aH
1ua6H	1uacH	1ub5A	1ub5H	1ub6H	1ub6A	1ucbH	1uj3B
1um4H	1um5H	1um6H	1uweV	1uweY	1uweH	1uwgY	1uwxM
1uwxH	1uywH	1uywM	1uz6F	1uz6P	1uz6H	1uz6W	1uz8B
1uz8H	1v7mI	1v7mH	1vfaB	1vfbB	1vgeH	1vpoH	1w72H
1w72I	1wcbH	1wcbB	1wejH	1wt5A	1wt5B	1xf2B	1xf2H
1xf3H	1xf3B	1xf4H	1xf4B	1xgpB	1xgqB	1xgrB	1xgtB
1xguB	1xgyH	1xgyI	1xiwH	1xiwD	1y0lB	1y0lF	1y0lD
1y0lH	1y18F	1y18D	1y18H	1y18B	1yecH	1yeeH	1yefH
1yegH	1yehH	1yeiH	1yejH	1yekH	1yjdH	1ynkH	1ynlH
1yqvH	1yy8D	1yy8B	1yy9D	1yylH	1yylR	1yymR	1yymH
1za6B	1za6F	1za6D	1zanH	1zeaH	1zlsH	1zluM	1zluH
1zlvM	1zlwH	1zlwM	1ztxH	1zwiA	25c8H	2a1wH	2a1wI
2a6dB	2a6dH	2a6iB	2a6jH	2a6jB	2a9mH	2a9mI	2aabH
2adgB	2adiB	2adjB	2aepH	2ai0I	2ai0J	2aj3D	2aj3B
2aj3F	2ajsH	2ajuH	2ajvH	2ajxH	2ajyH	2ajzB	2ajzH
2ak1H	2ap2A	2ap2C	2arjH	2arjB	2atkA	2b2xI	2b2xH
2bdnH	2bfvH	2bmkH	2bmkB	2bobA	2brrY	2brrH	2c1oH
2c1oB	2c1pB	2c1pH	2cgrH	2cjuH	2ck0H	2cmrH	2dblH
2dd8H	2ddqH	2dqCH	2dqdH	2dqeH	2dqfB	2dqfE	2dqgH
2dqhH	2dqiH	2dqjH	2dqtH	2dquH	2dwdA	2dweA	2eh7H
2eh8H	2eizB	2eksB	2f19H	2fatH	2fd6H	2fjfR	2fjfH
2fjfF	2fjfK	2fjfB	2fjfT	2fjfD	2fjfl	2fjfX	2fjfP
2fjfN	2fjfV	2fjgH	2fjgB	2fr4H	2fr4B	2fx7H	2fx8I
2fx8J	2fx8H	2fx8K	2fx9I	2fx9H	2g2rB	2g2rH	2g5bB
2g5bD	2g5bF	2g5bH	2g60H	2g75C	2g75A	2gcyB	2ghwD
2ghwB	2gjjA	2gjjB	2gjzB	2gjzH	2gk0B	2gk0H	2gkiB
2gkiA	2gsiH	2h1pH	2h8pA	2h9gB	2h9gH	2hfeA	2hfgH
2hg5A	2hh0H	2hjfA	2hkfH	2hkhH	2hmiD	2hrpH	2hrpN

2hvjA	2hvkA	2hwzH	2i5yH	2i5yR	2i60R	2ibzX	2igfH
2ih1A	2ih3A	2iptH	2ipuG	2ipuH	2iq9H	2iquH	2itdA
2j4wH	2j5lC	2jb5H	2jelH	2jk5A	2mpaH	2nljB	2nr6D
2nr6F	2nxyD	2nxzD	2ny0D	2ny1D	2ny2D	2ny3D	2ny4D
2ny7H	2nyyD	2o5xH	2o5yH	2o5zH	2ojzH	2ojzI	2ok0H
2op4H	2oqjK	2oqjH	2oqjE	2oqjB	2or9H	2or9I	2oslA
2oslH	2oz4H	2p7tA	2pcpB	2pcpD	2q76B	2q76D	2q8aH
2q8bH	2qqkH	2qqnH	2qscH	2r0lH	2r0wH	2r0zH	2r56H
2r56I	2rcsH	2uudJ	2uudH	2uylW	2uylB	2uylY	2uylN
2uziH	2v7hH	2v7nD	2v7nH	2vdkH	2vdlH	2vdmH	2vdmH
2vdoH	2vdpH	2vdqH	2vdrH	2vh5H	2vl5A	2vl5C	2vq1F
2vq1B	2vxqH	2vxtH	2vxuI	2vxuH	2w0fA	2w9dH	2wubR
2wucH	2xa8H	2xknB	2xknD	2xqyJ	2xqyG	2xraH	2xzaH
2xzcH	2xzqH	2y06H	2y07H	2y36H	2y5tA	2y6sD	2y6sH
2ybrA	2yk1H	2yklH	2ymxH	2yssB	2z4qB	2z91C	2z92A
2zchH	2zkhH	2zpkH	2zpkI	35c8H	3a67H	3a6bH	3a6cH
3aazA	3aazH	3auvB	3auvC	3b2uQ	3b2uJ	3b2uW	3b2uC
3b2uF	3b2uH	3b2uN	3b2uT	3b9kD	3b9kH	3baeH	3bdyH
3be1H	3bkcH	3bkjH	3bkmH	3bkyH	3bn9D	3bt2H	3bz4D
3bz4F	3bz4H	3bz4B	3c2aI	3c2aH	3c5sD	3c6sH	3c6sF
3c6sB	3c6sD	3cfbH	3cfbB	3cfcH	3cfdH	3cfdB	3cfeB
3cfeH	3cfjB	3cfjF	3cfjD	3cfjH	3cfkK	3cfkH	3cfkB
3cfkD	3cfkI	3cfkF	3cfkP	3cfkN	3cleH	3clfH	3cmoY
3cmoH	3cvhQ	3cvhH	3cviH	3cx5J	3cx5U	3cxdH	3cxhJ
3cxhU	3d85B	3d9aH	3detE	3detC	3dggB	3dggD	3difD
3difB	3dsfH	3durB	3durD	3dusD	3dusB	3duuB	3duuD
3dv4B	3dv6B	3dvgB	3dvnB	3dvnH	3ejzC	3ejzE	3eo0B
3eo0D	3eotH	3esvG	3esvF	3et9F	3eyfD	3eyfB	3eysH
3eyuH	3eyvB	3eyvH	3f58H	3fb5A	3fctB	3fctD	3fn0H
3fo0H	3fo1H	3fo1B	3fo2H	3fo2B	3fo9H	3fo9B	3fzuC
3fzuH	3g04B	3g5vB	3g5xD	3g5xB	3g5zB	3g6aH	3g6aB
3gb7A	3gbnH	3ggwD	3ggwB	3ghbH	3ghbI	3gheH	3gi8H
3gi9H	3gjeB	3gjeH	3gjfH	3gjfM	3gk8H	3gkwH	3gkzA
3gm0A	3grwH	3h3bC	3h3bD	3h42H	3haeI	3haeT	3haeO
3haeH	3hc0A	3hc0H	3hc3H	3hi1B	3hi1H	3hi5H	3hi6X
3hi6H	3hr5B	3hr5H	3hr5J	3hr5I	3hzkB	3hzmB	3hzbB
3hzyB	3i02B	3i02D	3i2cH	3i75B	3idxH	3ietB	3ietD
3if1B	3if1D	3ifoH	3ifoA	3ifpH	3ifpA	3ifpC	3ifpE
3igaA	3ijhD	3ijhB	3ijsD	3ijsB	3ijyD	3ijyB	3ikeD
3ikeB	3inuH	3iu3C	3iu3A	3iu3H	3ixtH	3ixtA	3juyB
3juyF	3juyE	3juyD	3juyA	3juyC	3jwdH	3jwdP	3k2uH
3kdmB	3kdmH	3klhD	3kr3H	3ks0K	3ks0H	3kymP	3kymH
3kymB	3kymJ	3kymF	3kymN	3kymL	3l5wB	3l5wH	3l5xH
3l5yH	3l7eH	3l7eB	3l7fH	3l7fB	3l7fE	3l95B	3l95H
3ld8C	3ldbC	3lexA	3lexH	3leyH	3lh2J	3lzfH	3ma9H

3macH	3mbxH	3mckB	3mckH	3mclH	3mj8B	3mj8H	3mj9H
3mlxH	3mlxI	3mlyI	3mlyH	3mlzH	3mnvB	3mnvD	3mnwB
3mnzB	3mo1B	3n9gH	3na9H	3naaH	3nacH	3ncjH	3nfpH
3nfpA	3nfsH	3ngbJ	3ngbE	3ngbB	3ngbH	3nh7K	3nh7H
3nh7J	3nh7I	3nidH	3nidE	3nifE	3nifH	3nigE	3nigH
3ntcH	3nz8A	3nz8H	3nzhH	3o11B	3o11H	3o2dH	3o2vH
3o2wH	3o41A	3o41H	3o45A	3o45H	3o6kH	3o6lH	3o6mH
3oauH	3oayM	3oayH	3oazM	3oazH	3ob0M	3ob0H	3ojdB
3okdB	3okeB	3okkB	3oklB	3oknB	3okoB	3or6A	3or7A
3pgfH	3phoB	3phqB	3pnwH	3pnwB	3pnwW	3pnwK	3pnwT
3pnwN	3pnwQ	3pnwE	3pp3I	3pp3H	3pp4H	3q1sH	3q3gH
3q3gD	3q3gB	3q3gK	3qctH	3qcuH	3qcuI	3qcvI	3qcvH
3qehG	3qehA	3qehE	3qg6B	3qg6H	3qg7H	3qhfH	3qosH
3qotH	3qpqH	3qpqD	3qpqJ	3qpqF	3qpxH	3qq9H	3qq9D
3qrgH	3qwoH	3qwoA	3r06B	3r06F	3r06D	3r06H	3rlgH
3ra7H	3ra7I	3rkdH	3rkdD	3ru8H	3rvtD	3rvuD	3rvvD
3rvwD	3rvxD	3s34H	3s35H	3s37H	3sgdJ	3sgdH	3sgeH
3sgeJ	3skjH	3skjI	3so3C	3sobH	3sqoH	3stlA	3stzA
3t2nH	3t2nI	3t3mH	3t3mE	3t3pE	3t3pH	3tnmH	3tnnC
3tnnE	3tnnH	3tnnA	3u0tB	3u0tD	3u0wH	3u30C	3u30F
3u7wH	3u7yH	3u9pK	3uc0H	3uc0I	3ujiH	3ujjH	3ujtI
3ujtH	3ulsH	3ulsB	3uo1H	3utzB	3ux9B	3ux9D	3uypA
3uyrH	3uzeA	3uzeB	3uzqA	3uzvB	3v52H	3v6fC	3v6fA
3v6fH	3v6fE	3v6oC	3v6oD	3vfgH	3vg0H	3vg9C	3vi3H
3vi3F	3vi4H	3vi4F	3vw3H	3w9dC	3w9dA	3w9eA	3wbdA
3wbdB	3we6A	3wfbH	3wfcH	3wfdH	3wfeH	3wfhA	3whxA
3wifA	3wihI	3wihH	3wiiH	3wiiI	3wkmH	3wkmI	3wxvH
3wxwH	3x3fH	3x3gH	3zdxE	3zdxH	3zdyH	3zdyE	3zdzE
3zdzH	3ze0H	3ze0E	3ze2E	3ze2H	3zkmH	3zkmC	3zknC
3zknH	3zl4H	43c9F	43c9H	43c9D	43c9B	43caH	43caF
43caD	4a6yH	4a6yB	4aehH	4aeiJ	4aeiH	4aeiI	4ag4H
4al8H	4alaH	4amkH	4at6H	4at6C	4at6E	4bh7B	4bh8B
4buhA	4bz1H	4bz2H	4c83C	4c83A	4cadE	4cadK	4cadB
4cadH	4cmhB	4cniA	4cniH	4d9lK	4d9lH	4d9lJ	4d9lI
4d9qE	4d9qH	4d9rE	4d9rH	4dgiH	4dgvH	4dgyH	4dkfI
4dkfH	4dn3H	4dn4H	4dtgH	4dvbH	4dvbA	4dvrH	4ebqH
4edwH	4edxH	4edxB	4eneE	4eowH	4ersH	4etqH	4etqA
4evnA	4evnO	4evnI	4evnG	4evnK	4evnM	4evnC	4evnE
4f33F	4f33H	4f33B	4f33D	4f37H	4f37F	4f3fB	4f58H
4f58K	4f58I	4f58J	4fabH	4ffvD	4ffvH	4ffwH	4ffwD
4ffyH	4fqhA	4fqhH	4fqih	4fqjH	4fqlH	4fzeH	4g3yH
4g5zH	4g6fB	4g6fH	4g6jH	4g6kH	4g6mH	4g7vH	4g7yH
4gagH	4gayH	4gmsI	4gmsJ	4gmsH	4gmtH	4gmtI	4gq9H
4gqpH	4gsdH	4gw1D	4gw1B	4gw5B	4gw5D	4gxvH	4gxvI
4h88H	4h8wH	4hbcH	4hc1H	4hc1M	4hcrM	4hcrH	4hdiH

4hdiB	4hfwH	4hfwB	4hh9D	4hh9B	4hhaB	4hieB	4hihB
4hiiB	4hixH	4hjbB	4hk0C	4hkzB	4hlzG	4hlzK	4hlzI
4hs6H	4hs6B	4hs8H	4hwbH	4hweH	4hxaH	4hxbH	4hziH
4hziA	4i2xB	4i2xD	4i77H	4i9wG	4i9wE	4ij3C	4imkB
4imkA	4imlA	4imlH	4ioiB	4irzH	4j1uD	4j4pH	4j4pC
4j6rH	4j8rB	4j8rD	4jamH	4jdvH	4jdvA	4jfxB	4jfxH
4jfyH	4jfyB	4jzH	4jg0H	4jg1H	4jhaH	4jkhH	4jlrA
4jlrH	4jn1H	4jn2H	4jpiH	4jpkH	4jpwH	4jqihH	4jr9H
4jreB	4jreH	4jzjH	4jzjA	4jznI	4jzoD	4jzoA	4jzoG
4jzoC	4k23H	4k2uH	4k2uI	4k3jH	4k7pH	4k7pY	4k94H
4k9eH	4kaqH	4khtH	4khxH	4ki5C	4kjqE	4kjqC	4kk8C
4kk8E	4kk9C	4kk9E	4kklC	4kklE	4kmtH	4kphI	4kphH
4kq3H	4kq4H	4krpD	4kucH	4kucF	4kuzH	4kvcH	4kxzH
4kxzJ	4kxzQ	4kxzN	4ky1H	4kzdH	4kzeH	4l5fH	4laqH
4larH	4lasH	4lbeA	4lciH	4lcuA	4leoA	4lexH	4lf3B
4lf3E	4liqH	4llcB	4llxA	4lluC	4lluA	4llvH	4llwC
4llyA	4llyC	4lmqE	4lmqH	4louC	4louE	4lqfH	4lssH
4lstH	4lsuH	4lu5I	4lu5H	4lvhH	4lvhB	4lvhE	4lvnC
4lvoC	4m1dI	4m1dH	4m1gH	4m43H	4m48H	4m5yI	4m5yH
4m5zH	4m61B	4m61D	4m62I	4m62H	4m6mH	4m6nH	4m6oH
4m7jH	4m7kH	4m7zH	4m7zB	4m93B	4m93H	4ma1E	4ma1H
4ma1B	4ma7H	4ma8H	4mauH	4mswA	4n8cH	4n8cI	4n9gA
4n9gH	4n9gM	4nccH	4ncc2	4nghH	4nhcH	4nikB	4njaH
4nkiH	4nm4H	4nm4I	4nnpH	4nnpX	4np4I	4np4H	4nwtH
4nx3B	4nylC	4nylH	4nylE	4nztH	4o51F	4o51D	4o51B
4o51H	4o9hH	4ocxH	4ocyH	4odsH	4odtH	4oduA	4oduH
4odvH	4odwA	4odwH	4ogxH	4ogyH	4ojfH	4okvA	4okvC
4oluH	4olvH	4olwH	4olxH	4olyH	4olzH	4om0H	4om1H
4onfH	4otxH	4otxI	4ouuB	4ouuH	4p3cH	4p3dA	4p3dH
4pb0H	4pb9H	4pljH	4pljD	4pozD	4pp2D	4pp2B	4ps4H
4pttB	4ptuB	4py7A	4py7I	4py8I	4q0xH	4q2zH	4q2zA
4qciH	4qciB	4qf1A	4qf1H	4qhuH	4qhuB	4qt5B	4qt5H
4qthA	4qthH	4qwwF	4qwwD	4qygH	4qxtA	4qxuH	4qy8A
4qyoA	4r3sA	4r7dE	4r7dC	4r7dA	4r7dG	4r7dK	4r7dO
4r7dM	4r7dI	4r8wH	4r90H	4r97C	4ravA	4ravC	4rbpH
4rbpM	4rdqO	4rdqG	4rdqI	4rdqM	4rdqK	4rfeA	4rfeC
4rfeH	4rfeE	4rgmH	4rgmC	4rgnF	4rgnH	4rgnD	4rgnB
4rgoH	4rrpG	4rrpK	4rrpH	4rrpI	4rrpL	4rrpJ	4rwyH
4rzcC	4rzcE	4rzcH	4s1dO	4s1dE	4s1dC	4s1dH	4s2sA
4s2sH	4tprH	4tqeH	4tsaH	4tsbH	4tscH	4ttcD	4ttcH
4tukH	4tuoA	4tuoC	4u0rB	4u6vK	4u6vH	4uikH	4uilH
4uimE	4uimC	4uimA	4uimH	4uinH	4uu9A	4uu9H	4uv7H
4wcyH	4webH	4wfeG	4wfeE	4wffE	4wffG	4whtC	4whtQ
4whtK	4whtM	4whtE	4whtI	4whtU	4whtA	4whtG	4whtX
4whtS	4whtO	4whyM	4whyI	4whyG	4whyK	4wukH	4wv1E

4wv1B	4wy7H	4x0kA	4x0kH	4x4xA	4x4yA	4x7sH	4x80H
4xakD	4xakH	4xbgH	4xbgA	4xbgE	4xbpH	4xbpC	4xbpE
4xc1H	4xc3H	4xccH	4xceC	4xceA	4xcnH	4xcnE	4xcnA
4xcnC	4xgzJ	4xgzM	4xgzQ	4xgzW	4xgzH	4xgzO	4xgzC
4xgzA	4xgzU	4xgzE	4xgzS	4xgzG	4xmlH	4xnmB	4xnmH
4xnqH	4xnqB	4xnuH	4xp9H	4xpaH	4xphH	4xrcB	4xtrC
4xtrE	4xvjH	4xvsH	4xvtH	4xvuE	4xvuI	4xvuK	4xvuC
4xwgH	4xwoE	4xwoK	4xwoI	4xwoC	4xwoW	4xwoO	4xwoQ
4xwoU	4xxdB	4xxdE	4xzuA	4y5vA	4y5vD	4y5vG	4ydkH
4ygvF	4ygvA	4yhiA	4yhiH	4yhkH	4yhlH	4yhmH	4yhnF
4yhnA	4yhoH	4yhpH	4yhpE	4yhpC	4yhpA	4yhyH	4yhyB
4yhzH	4yjlL	4yk4Z	4yk4C	4yr6D	4yr6A	4ywgI	4ywgH
4yxhH	4yxkH	4yxlH	4z0xB	4z8fH	4z95H	4zd3H	4zffA
4zffH	4zfgH	4zfoH	4zfoA	4zptA	4zs7H	4zsoD	4zsoB
4zykA	5a16A	5a16E	5a16C	5a16G	5a2iH	5a2jH	5a2kH
5a2lH	5aamB	5aamA	5albH	5anmD	5anmB	5aumA	5aumH
5awnH	5ayuH	5az2H	5b3jE	5b3nA	5b4mH	5b4mC	5b71D
5b71B	5b8cK	5b8cH	5b8cE	5b8cB	5bjzC	5bjzD	5bk1C
5bk1H	5bk2H	5bk2C	5bk3H	5bk3C	5bk3E	5bk3A	5bo1H
5bo1I	5bv7H	5bv7C	5bvjF	5bvjD	5bvjB	5bvjH	5bvpH
5bzdD	5bzdB	5bzwB	5c2bH	5c6wJ	5c6wH	5c7xH	5cbaA
5cbaC	5cbeC	5cbeA	5cckH	5cd3C	5cd3A	5cd3E	5cd3G
5cgyC	5cgyH	5chnH	5cilH	5cinH	5cipH	5cipA	5cmaB
5cp3H	5czxH	5czxC	5d72H	5d7sH	5d93C	5d93F	5d96J
5d96C	5dd0H	5dd1H	5dd3H	5dd5H	5dfvE	5dfvC	5dfwH
5dhvH	5dhvA	5dhxB	5dhxL	5dk3B	5dk3G	5dlmH	5dlmI
5do2H	5do2C	5dq9A	5dq9C	5dq9H	5dqdH	5dqjA	5dqjH
5dr5H	5drnA	5drnH	5drwA	5drxH	5drxA	5drzB	5drzH
5ds8A	5ds8H	5dscA	5dscC	5dscE	5dscH	5dtfA	5dtfH
5dubH	5dubA	5durH	5durB	5dyoA	5dyoH	5e08H	5e2tH
5e2vH	5e2wH	5e8dH	5e94D	5e94B	5ea0H	5eblA	5ebmA
5ebwA	5ec1A	5ec2A	5eiiH	5eiiA	5en2A	5eocH	5eocJ
5eoqH	5epmA	5epmE	5erwA	5esaA	5esqD	5esqB	5etuB
5etuD	5eukD	5eukB	5ewiH	5eziH	5ezjA	5ezlA	5ezlH
5f3bA	5f3bE	5f3hE	5f3hG	5f3hC	5f3hA	5f6hK	5f6hM
5f6hI	5f6hO	5f6iB	5f72T	5f72S	5f7eH	5f88D	5f88B
5f96H	5f9oH	5f9wB	5f9wH	5fb8B	5fcsL	5fcsH	5fcuH
5ff6B	5ff6D	5fgcE	5fhaH	5fhbH	5fuzH	5fxcH	5ggqH
5ggsA	5ggsC	5ggtH	5gguA	5gguH	5ggvH	5ghwH	5girH
5girA	5gisH	5gjsH	5gkrA	5gksA	5gksC	5gruL	5gruH
5grwB	5grwA	5grxH	5grxG	5gryA	5gryG	5grzA	5grzG
5gs1D	5gs1F	5gs1C	5gs1L	5gs1J	5gs1G	5gs1H	5gs1N
5gs1K	5gs1R	5gs1E	5gs1B	5gs1P	5gs1I	5gz0D	5gz0B
5gzoH	5gzoC	5h35H	5h35F	5h35A	5hbtD	5hbvD	5hdqH
5hpmD	5hpmB	5hyqB	5hyqD	5hysC	5hysH	5hysE	5hysA

5i15H	5i16H	5i16B	5i18H	5i19H	5i1aH	5i1aB	5i1cH
5i1dH	5i1eH	5i1gH	5i1hH	5i1iH	5i1kH	5i1lB	5i1lH
5i2iD	5i2iB	5i30H	5i66A	5i76D	5i8cA	5i8eA	5i8eC
5i8kH	5i8oH	5ibtH	5ibuG	5ibuH	5icxD	5icxB	5icyB
5icyD	5iczD	5iczB	5id0B	5id0D	5id1D	5id1B	5ifhH
5iieA	5iieH	5ijkA	5ijkB	5ik3C	5ikcB	5ikcH	5iopD
5iopB	5iq9A	5iq9H	5ir1B	5ir1D	5it2H	5itbH	5itfD
5itfB	5iv2B	5iv2D	5ivzD	5ivzB	5iwlA	5iwlB	5j13C
5j74A	5j74B	5j75A	5j75B	5j9pA	5jo4C	5jq6H	5jr1H
5jrpH	5jueH	5jw5H	5jxeD	5jylD	5jylB	5jymB	5jymD
5k59H	5k8aH	5k8aF	5k8aB	5k8aV	5k9jH	5k9kA	5k9kH
5kjrH	5kmvE	5kmvH	5kn5D	5kn5A	5kvdH	5kveH	5kvfH
5kvgH	5kw9H	5kzpG	5kzpE	5kzpH	5kzpF	5l0qC	5l0qF
5l6yH	5l7xH	5l88H	5l9dH	5lbsH	5lbsI	5ldnH	5lghH
5lqbH	5lspS	5lspH	5lwyH	5lx9H	5lxgH	5m63M	5m63H
5mesH	5mevH	5mheH	5mhgH	5mi0B	5mo3H	5mo9H	5mp3H
5mp3A	5mp5C	5mp5H	5mp5E	5mp5A	5mthH	5mthA	5mu0K
5mu0M	5mu0I	5mu0G	5mu0A	5mu0O	5mu0E	5mu0C	5mu2K
5mu2A	5mu2E	5mu2M	5mu2I	5mu2G	5mu2C	5mu2O	5muqH
5muqA	5mv3C	5mv3F	5mv3R	5mv3O	5mv3L	5mv3I	5mv3A
5mv3U	5mv4U	5mv4C	5mv4F	5mv4R	5mv4L	5mv4O	5mv4A
5mv4I	5mvjH	5mvjE	5mvjC	5mvjA	5mvzH	5mvzA	5mx3A
5mx3H	5my4B	5mykB	5myoD	5myoB	5myxB	5myxD	5n2kH
5n2kF	5n2kL	5n2kN	5n2kD	5n2kJ	5n2kB	5n2kP	5n4gH
5n4jH	5n7bH	5n7wH	5n7wA	5nbiH	5nbwB	5nbwH	5ngvH
5nh3I	5nh3H	5nhwH	5nivB	5nmvH	5nphH	5npiA	5npiB
5npjA	5npjB	5nuzH	5nuzA	5nyxH	5nyxM	5o14C	5o14H
5o1rI	5o1rH	5ob5H	5obfH	5ocaH	5occH	5ockH	5ocxH
5ocyH	5od0H	5od8A	5odbA	5ogiB	5otjB	5otjH	5owpH
5sx4H	5sx4J	5sx5H	5sx5J	5sy8H	5szfH	5t1kB	5t1kD
5t1lB	5t1lD	5t1mB	5t1mD	5t29H	5t4zA	5t4zH	5t5bH
5t5bA	5t5fH	5t6lA	5t6lH	5t6pB	5t6pD	5t78D	5t78B
5t80H	5t85H	5tbdA	5tbdJ	5tbdD	5tbdE	5tdnD	5tdoD
5tdpD	5te4H	5te6H	5te7H	5tf1H	5tfsH	5tfwH	5tgbH
5th2D	5th2B	5th9J	5th9I	5th9H	5tihH	5tkjG	5tkjA
5tkjD	5tkjJ	5tkkH	5tl5H	5tlkB	5tlkF	5tlkD	5tlkH
5tplH	5tppH	5tqaA	5tqaH	5tr8H	5trpH	5tz2H	5tztH
5tzuH	5u15A	5u15H	5u3dB	5u5fB	5u5mB	5u6aB	5ubyH
5ucbH	5ueaA	5ueaH	5uekH	5uelH	5uemH	5ugyJ	5ugyH
5ugyI	5uknH	5ukoH	5ukqH	5ur8A	5ushD	5ushH	5usiH
5usiA	5uslH	5uslB	5v7rH	5v7uH	5vagC	5vcoD	5vcoB
5vebA	5vebH	5vgaB	5vh3H	5vh3A	5vh4A	5vh4H	5vjoC
5vjoA	5vjqG	5vjqE	5vjqA	5vjqC	5vk6A	5vkdH	5vkeA
5vkhA	5vkkH	5vlpH	5vpgD	5vphD	5vplD	5vr9H	5vshE
5vshA	5vshH	5vshC	5vsiH	5vtaH	5vtaF	5vtaI	5vxrH

5vyfB	5vyfH	5vzxH	5vzxE	5vzyH	5w05H	5w06H	5w2bH
5w3eE	5w3lE	5w3mE	5w3pH	5w5xH	5w5zH	5w6cH	5w6gH
5wawH	5wb9H	5wcaH	5wcdH	5weqB	5whjH	5whkH	5wi9H
5wi9F	5wk2H	5wk3S	5wk3U	5wk3Q	5wk3W	5wkzH	5wl2A
5wn9H	5wnaC	5wnaH	5wnbI	5wt9H	5wtgD	5wtgB	5wttH
5wttA	5wuvH	5wuxA	5wuxH	5wuxC	5wymA	5wymC	5x0tC
5x0tA	5x4gA	5x5xH	5x8mB	5xajE	5xajF	5xajH	5xajA
5xcqA	5xcrA	5xcrD	5xcsA	5xcuA	5xcuD	5xcxA	5xhgD
5xhgB	5xj4H	5xkuC	5xlid	5xlic	5xmhD	5xmhH	5xrqA
5xrqH	5xwdH	5xxyH	5y11A	5y2kA	5y2kE	5y2lI	5y9jH
5y9kH	5yaxA	5yaxB	5yd3C	5yd3E	5yd3G	5yd3A	5yd4C
5yd4A	5yd4G	5yd4E	5yd5C	5yd5A	5ye3B	5ye4A	5ye4C
5yoyH	5yoyG	5yoyQ	5yoyR	5yoyP	5yoyI	5yslG	5yslE
5yslA	5yslC	5yupH	5yy4A	5yy5C	5yy5H	5ziaA	5ziaO
5ziaK	5ziaD	5ziaG	5ziaH	5zv3H	6a0xC	6a0xA	6a0xG
6a0xE	6a0zH	6a3wJ	6a3wD	6a3wG	6a3wA	6a67H	6a67C
6a76H	6a77H	6a78H	6a78I	6a79I	6a79H	6a9kB	6a9kH
6ad7C	6ad7E	6adbE	6al4F	6al4D	6al4B	6anaH	6aniI
6aniH	6aodB	6apcH	6aq7H	6arpB	6arpD	6au5D	6au5B
6avnH	6axkA	6axlA	6axlD	6axpB	6axpD	6aynD	6aynB
6ayzB	6ayzR	6az2A	6az2C	6azkD	6azkB	6azlB	6azlD
6azmD	6azmB	6azxC	6azxA	6azzC	6azzF	6b08C	6b0aH
6b0gD	6b0hB	6b0hD	6b0sH	6b0wH	6b14H	6b3kH	6b3sF
6b3sJ	6b3sH	6b3sC	6b5lH	6b5mH	6b5nH	6b5oH	6b5pH
6b5rH	6b5sH	6b5tH	6b9jA	6b9yB	6b9zB	6baeB	6bahB
6bb4I	6bb4J	6bb4H	6bckH	6bf4H	6bf4B	6bfqC	6bfqA
6bfqE	6bfqH	6bfsH	6bgtB	6bhzH	6bi0H	6bi2H	6bi2I
6bitJ	6bitI	6bkcH	6blhH	6bliJ	6bliA	6bliD	6bliG
6bpaB	6bpaE	6bpbA	6bpcB	6bpcE	6bqbH	6by2A	6by3A
6bzuC	6bzuA	6bzuG	6bzuE	6bzvA	6bzvE	6bzvC	6bzvG
6bzwE	6bzwC	6bzwG	6bzwA	6bzyH	6c5hH	6c5iH	6c5jH
6c5kH	6c5kA	6c6xH	6c6xA	6c6xE	6c6xC	6c6zH	6c6zC
6c9uH	6cbjH	6cbpA	6cbvH	6cdmA	6cdmD	6cdoA	6cdpA
6cezH	6cnrH	6co3H	6cr1H	6crkH	6ct7A	6ct7H	6cvkA
6cvkB	6cw2A	6cw3C	6cw3A	6cxgM	6cxgH	6cyfD	6cyfU
6cyfF	6cyfM	6d0xA	6db5I	6db5H	6db6H	6db8H	6db9H
6dc4H	6dc7I	6dc7H	6dc8H	6dc9I	6dcaH	6dcaI	6dcaK
6dcaJ	6dcvB	6dcvH	6dcwH	6ddmB	6ddrB	6ddvB	6df0H
6df1H	6df2H	6df2B	6dfiH	6dfjH	6dg2C	6dg2A	6dkjH
6dkjA	6dlaH	6dlbH	6dn0A	6dsiA	6dw2B	6dwiL	6dwiJ
6dwiB	6dwiD	6dwiH	6dwiN	6dwiP	6dwiF	6dznH	6dzrH
6e4xZ	6e56H	6e56G	6ehvX	6ehwB	6ehxB	6ehyA	6ejgD
6ejgC	6ejmI	6ejmH	6ek2H	6ek2I	6eleH	6eljA	6eljH
6ellH	6eluH	6elub	6eluE	6eluK	6emjH	6erxH	6erxF
6erxQ	6erxM	6erxD	6erxO	6ev2A	6ev2C	6ev2G	6ev2E

6ewbH	6ewbE	6ewbJ	6ewbG	6eynD	6fabH	6faxH	6fbjH
6ffjC	6ffjA	6fg1H	6fg1A	6fg2F	6fg2H	6fg2D	6fg2I
6fjbH	6flaA	6flaH	6flbH	6flcH	6flcB	6foeH	6foeA
6frjH	6fxnQ	6fxnM	6fxnH	6fxnF	6fxnO	6fxnD	6fy0H
6fy0I	6fy2X	6fy2H	6fzqH	6fzrH	6g8rA	6gc2H	6gfeK
6gfeH	6ghgA	6ghgH	6gk7H	6gk8H	6gkuH	6glwH	6glwC
6gv4H	6gxxH	6h06C	6h06E	6h06A	6h06H	6h0eH	6h0eC
6h0eE	6h0eA	6h2yH	6h3hA	6h3hC	6h3tI	6h3tH	6hf1C
6hf1F	6hgaH	6hguA	6hguH	6hhcH	6hkgC	6hkgA	6hx4H
6hx4I	6hxcW	6hxcH	6i07A	6i07B	6i1oH	6i8sE	6i8sF
6i8sG	6i8sH	6iapH	6iapE	6iasH	6idgH	6idhH	6iebH
6iebE	6iguH	6ionH	6iuvC	6iuvH	6ivzH	6iw2N	6iw2K
6j5dH	6j5fH	6j71B	6j9oH	6jbtH	6jc2H	6jjpD	6jjpA
6ju0I	6ju0J	6ju0H	6ju0K	6jwcH	6k0yA	6k41H	6k4zA
6k4zB	6k65H	6kdiH	6kpfS	6kr0C	6kr0B	6kr0D	6kr0A
6ks0H	6ks1H	6ks2B	6ks2D	6ktrF	6ktrA	6kx0A	6kx1A
6kyzB	6kyzE	6kz0E	6kz0B	6kz0K	6kz0H	6l8tH	6l8tA
6l8tC	6lgwC	6lktA	6lz4D	6lz4A	6lz9H	6m3bC	6m58C
6m58H	6m87L	6m87F	6m87J	6m87B	6m87H	6m87D	6me1A
6me1C	6medG	6medC	6medA	6meeC	6megH	6mehH	6meiH
6mejA	6mejH	6mfjH	6mfjA	6mftH	6mftA	6mg7H	6mhrD
6mhrA	6mi2A	6mi2D	6mlkH	6mnfM	6mnfH	6mnqH	6mnrH
6mnsA	6mnsH	6mqcA	6mqeH	6mqeA	6mqrH	6mqsa	6mqsc
6msyH	6mtoH	6mtpH	6mtpA	6mtqH	6mtrH	6mtsH	6mttH
6mtxH	6mtyH	6mu3M	6mu3H	6mubM	6mubH	6mwnC	6mwnH
6mxrH	6mxrA	6mxsH	6mxsA	6my4H	6my4A	6my5A	6my5H
6n16C	6n16A	6n16H	6n16K	6n32K	6n32H	6n35M	6n35H
6n6bK	6n81C	6n81H	6nb8H	6nfnJ	6nfnN	6nfnF	6nfnB
6nfnD	6nfnH	6nfnR	6nfnP	6nfuA	6nfvA	6nhaH	6nisH
6nmrA	6nmrH	6nmrJ	6nmrF	6nmsH	6nmsB	6nmuB	6nmuH
6nouA	6novA	6novC	6nuyH	6nuzH	6nv0D	6nv0H	6nyqH
6o23A	6o23C	6o24A	6o26A	6o28C	6o28A	6o29A	6o29H
6o2cA	6o39B	6o3aB	6o3bG	6o3bB	6o3dH	6o3kH	6o3lH
6o3lA	6o41H	6o41B	6o41D	6o42B	6o42H	6o9hH	6o9hA
6o9iD	6o9iA	6oanB	6oanD	6obdH	6obdB	6obzH	6obzA
6oc3C	6oc3A	6ocbH	6ogxC	6ogxH	6ohgC	6okmH	6ol5A
6ol5H	6ol6H	6ol6A	6ol7H	6ol7E	6ol7M	6ol7P	6ol7K
6ol7I	6ol7A	6ol7C	6oshH	6osvH	6otcH	6oy4D	6oz2H
6p4aH	6p4bH	6p4cH	6p60G	6p60J	6p60A	6p60C	6p79H
6p7hA	6p8dA	6p8dD	6p9iA	6p9iC	6p9jH	6pa0A	6pbvB
6pbvD	6pdrH	6pdsB	6pdsH	6pduH	6pe7H	6pe8A	6pe8H
6pecH	6pefA	6pefD	6phbB	6phbD	6phcC	6phcA	6phhA
6phhC	6pi7C	6pi7F	6pilA	6pisH	6pisI	6pk8D	6pk8B
6pk8A	6pk8C	6plhB	6plkI	6plkH	6ppgB	6ppgH	6pxgE
6pxgH	6pxgA	6pxgC	6pxhC	6pxrH	6pzeH	6pzfF	6pzfH

6pzgH	6pzhH	6pzhB	6qb3X	6qb4X	6qb6H	6qb9A	6qb9X
6qbcH	6qcuH	6qf6B	6qf6A	6qf6C	6qf6D	6qf9A	6qf9B
6qfcB	6qigH	6qnkB	6qnkD	6qu9A	6qu9H	6r8xC	6rcoC
6rcoA	6rcqA	6rnnD	6rnnH	6rp8h	6rp8H	6rpsH	6rpsN
6s5aH	6s8iP	6s8iH	6s8iY	6s8jP	6s8jH	6s8jY	6sf6A
6sf6H	6suzH	6svlH	6svlG	6svlO	6svlD	6svlK	6svlA
6sxiH	6sxiA	6tcmH	6tcnH	6tcnB	6tcoB	6tcoH	6tcpH
6tcpB	6tcpD	6tcpF	6tcsA	6tggH	6touA	6tybH	6u1tH
6u2fH	6u36H	6u38H	6u3iH	6u3zC	6u3zA	6u6uH	6u85H
6u8dH	6u8kH	6u8kF	6u8kD	6u9sD	6u9sA	6u9uH	6ubiD
6ubiA	6uceH	6ucfH	6ug7H	6ug8H	6ug9A	6ug9J	6ug9H
6ugaB	6ugaH	6ugsH	6ugsA	6ugtH	6ugtA	6uguH	6uguA
6ugvA	6ugvH	6uleC	6uleA	6ulfA	6umxH	6umxh	6uoeH
6urmD	6urmH	6uteI	6uteG	6uteE	6uteA	6uteC	6uudH
6uvoH	6uynH	6v4nA	6v4nD	6v4nC	6v4nH	6v4pC	6vboH
6vbpA	6vbpC	6vbpJ	6vbpH	6vbqH	6vbqA	6vbqC	6vbqE
6velH	6vepU	6vepO	6vepI	6vepC	6vgrH	6vgrC	6vjtH
6vl8H	6vl8E	6vl8C	6vl8A	6vl9A	6vl9H	6vlnH	6vn1J
6vn1I	6vn1H	6vorC	6vosH	6vpyA	6vpyH	6vtwH	6vu2H
6w00H	6w05H	6w4vC	6wbvH	6wdsH	6wfyH	6wfhH	6wfhA
6wg0H	6wg1H	6wg1A	6wg2I	6wg2H	6wg8A	6wgjE	6wgkA
6wgkE	6wgkC	6wglA	6wh9E	6wh9B	6wh9H	6wioA	6wirA
6wiyH	6wj0H	6wmwH	6wmwM	6wn4A	6wn4H	6wo4H	6wo5H
6wo5G	6wt3H	6wyrI	6wytH	6x3tK	6x3tJ	6x3xJ	6x3xK
6x40J	6x40K	6x5eB	6x5eH	6x5mh	6x5mH	6xc3H	6xc3B
6xc4X	6xc4H	6xc7H	6xc7C	6xcaH	6xe1H	6xkpH	6xkpM
6xkqH	6xkrH	6xliH	6xliA	6xliC	6xxvD	6xxvA	6xzwH
6y1kH	6y1lH	6y1nH	6y54H	6y54M	6y54O	6y9bI	6y9bJ
6y9bH	6ylaH	6ylaB	6ywcA	6ywcD	6z2lC	6z2mB	6z2mH
6z3kQ	6z3kM	6z3kJ	6z3kG	6z3kO	6z3kC	6z3kA	6z3kH
6z3kE	6z3pH	6zczH	7bwjH	7bz5H	7c01H	7c01C	7c94B
7c94E	7c95D	7c95B	7ch5H	7chbH	7chcH	7chcA	7chfH
7chfA	7cu5A	7cu5B	7jmoH	7jmpH	7jmwH	7jn5H	7jooH
7jx3A	7jx3H	7k3qH	7k43F	7k43H	7k43C	7k8oH	7k8pH
7k8qH	7k8rH	7k9zH	7k9zB	8fabD	8fabB		

Table A.3: Train and validation sets used for ABlooper (see Chapter 2). The first four characters form the PDB ID, the final character is the heavy chain ID.