Explainable Deep Generative Models, Ancestral Fragments, and Murky Regions of the Protein Structure Universe: Datasets, Models, and Analyses of Fold Space

Eli Jacob Draizen

Charlottesville, VA

A Dissertation submitted to the Graduate Faculty
of the University of Virginia in Candidacy for the Degree of
Doctor of Philosophy

Department of Biomedical Engineering

University of Virginia

December 8th 2022

Explainable Deep Generative Models, Ancestral Fragments, and Murky Regions of the Protein Structure Universe: Datasets, Models, and Analyses of Fold Space

Eli Jacob Draizen

(ABSTRACT)

Modern proteins did not arise abruptly, as singular events, but rather over the course of at least 3.5 billion years of evolution. Can machine learning teach us how this occurred? The molecular evolutionary processes that yielded the intricate three-dimensional (3D) structures of proteins involve duplication, recombination and mutation of genetic elements, corresponding to short peptide fragments. Identifying and elucidating these ancestral fragments is crucial to deciphering the interrelationships amongst proteins, as well as how evolution acts upon protein sequences, structures & functions. Traditionally, structural fragments have been found using comparative approaches such as sequence alignment and 3D structural superposition, but that becomes challenging when proteins have undergone extensive permutations—allowing two proteins to share a common architecture, though their topologies may drastically differ (a phenomenon we term the *Urfold*). In my thesis, I develop several tools and datasets, leveraging decades worth of structural biology knowledge in light of the Urfold model of protein structure, in order to decipher the underlying molecular bases for protein structural relationships.

- For my first aim, I developed a community resource to create and share protein properties–structural, biophysical and evolutionary—for utilization in structural bioinformatics pipelines that involve machine learning. These properties can be used as feature-sets in any machine learning model; besides reusability and efficiency, such a resource would also facilitate more reproducible workflows, by ensuring analyses are performed with standardized data. This project, termed 'Prop3D', is described in Chapter 2. The work, which has been written-up for submission to a journal in December 2022.

- In my second aim, I designed a sequence-independent, alignment-free, rotationally-invariant similarity metric of protein inter-relationships based on Deep Generative Models and 3D structures. Motivated by the Urfold view of protein structure, this framework leverages similarities in latent-spaces rather than the 3D structures directly, and it encodes biophysical properties; this capability, in turn, allows higher orders of similarity to be detected among proteins that are presumed to be only distantly related. I used this new similarity metric to detect clusters, or 'communities', of similar protein structures using Stochastic Block Models. This method takes a rather different approach to traditional clustering, allowing for proteins to span multiple clusters, thereby more explicitly allowing for the continuous nature of fold space. This project, termed 'DeepUrfold', is described in Chapter 3. The work, which was submitted to Bioinformatics in November 2022, is also available as a preprint at https://doi.org/10.1101/2022.07.29.501943.

- Finally, for my last aim, I sought to discover if particular residues/peptide fragments from a given domain might be responsible for conferring the similarity/linkage to other domains—including those relationships which may be exceedingly remote—using Layer-wise Relevance Propagation, an Explainable AI technique. This in turn creates an automatable/systematic and reproducible framework to identify new urfolds across the protein structure universe. This project, termed ‘DeepUrfold-explain’, is described in Chapter 4. Though somewhat nascent, the project has been accepted to the peer-reviewed Machine Learning in Structural Biology (MLSB) workshop at the Neural Information Processing Systems (NeurIPS) conference held in December 2022, and is also currently available as a preprint at https://doi.org/10.1101/2022.11.16.516787.

Dedication

I dedicate my dissertation to my mom, dad, and sister for supporting me and encouraging me all of the way through.

Acknowledgments

I would like to thank my advisors Phil Bourne and Cam Mura for supporting during this entire process — I would not have been able to finish my dissertation without their guidance, feedback and encouragement — and I would like to thank my committee for giving invaluable advice. I would also like to thank David Landsman & Anna Panchenko for support and motivation before I started my PhD, up until I transferred to UVA. Dietlind Gerloff was my academic advisor starting from my time at UC Santa Cruz and I have always valued her advice, input and support; I would also like to thank her for support early on during my PhD.

I would also like to thank all of my friends from Boston, DC, Charlottesville, and around the world. From the NIH Graduate Student Underground, to my friends and housemates in Charlottesville, to ISMB Conferences where I met lifelong friends. The United Campus Workers Virginia union also played an important role during my PhD, so I would like to thank all of the other PhD student and employees for their solidarity.

Music has been my primary way to get through the PhD. I would like to thank all of my friends in the bands I have played with and those who helped me run DIY music venues:

- Aly Gorey (Band in Boston 2015-2016 w/ Ethan Hoffman; https://alygoreyjd.bandcamp.com)
- Dothraki Decepetion & Marble House (Band and venue respectively in DC 2016-2018 w/ friends from the NCBI: Alexander Goncearenco, Guilhem Faure, Nicolas Fiorini and Sergey Shmakov; https://soundcloud.com/dothrakideception)

- The Ducklettes (Band in DC 2017-2018 w/ friends from NIH: Kristoffer Johansen, Dan Konzman, Anna-Leigh Brown and Mike Tisza)
- Orange Folder (Band in Charlottesville 2018-2020 w/ Max Hoffman, Jack Richardson, Corrine James and Thomas Dean; https://orangefolder.bandcamp.com)
- Chinchilla Cafe (Venue in Charlottesville w/ Robin Brown, Lane Rasberry, Fabian Garcia, Gabriela Toledo and Laura Fontenas; https://commons.wikimedia.org/wiki/Category:Chinchilla_Caf%C3%A9)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Proteins are key biological macromolecules that perform most of the physiological functions of cellular life. Each protein has a specific 3D structure, which generally defines the protein's function. Such functions can include transcription of DNA, breaking down chemicals, and passing signals throughout the cell by interacting with other proteins. The diversity of these 3D shapes and functions make all life possible.

Protein function arises from sequence, structure, and evolution. Proteins with similar amino acid sequences, 3D shapes, and evolutionary histories often share many functional properties, such as protein-protein interactions and small-molecule (drug) interactions; at that scale, the entire protein structure, or even just small segments, can be similar even between proteins that are only distantly related. Understanding protein function is important in all realms of biology. For example, in drug discovery knowing a protein's structure and function helps in identifying drug targets and off-target drug interactions, and understanding how a drug will affect a metabolic system by binding to its protein target, potentially disrupting important biological pathways. If we know the structure and function of one protein, such as the binding properties of a specific drug compound, we may be able to accurately 'transfer' the annotation to a new protein of interest, given sufficient similarity.

Most biomedical questions concern present-day proteins, so-called 'extant' proteins that are encoded in the genomes of modern living organisms (e.g. humans) with specific functions occurring at this point in time. However, peering into the past to understand how extant proteins have evolved gives clues as to both biochemical functions and mechanistic properties. The ability to successfully transfer functional annotations hinges on the fact that all proteins are related throughout evolution. High-throughput methods to transfer function annotation across the set of all possible proteins, called the "protein universe", involves organizing proteins into groups of related proteins.

The precise historical trajectory of the protein universe, in terms of primordial peptides, protein domains, and multi-domain proteins remain unknown. We know very little about what life was like before the Last Universal Common Ancestor (LUCA) [1]. This thesis considers patterns of similarity and interrelationships across the protein universe with an emphasis on protein structure to give clues about times pre-LUCA. Protein structure provides the natural bridge between sequence and function and will be the primary source of data in this thesis. Overall, my thesis can be seen as an attempt to further understand the origin of life.

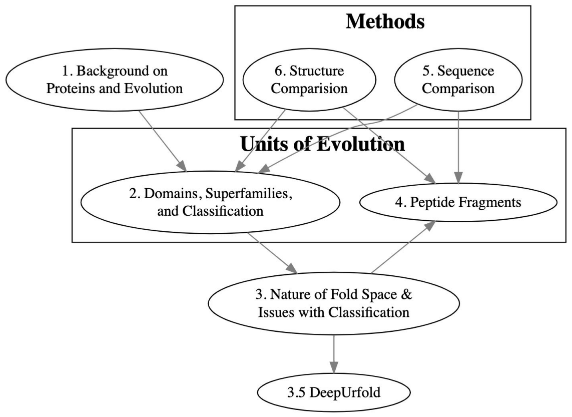

Methods
1. Background on Proteins and Evolution
6. Structure Comparision
5. Sequence Comparison
Units of Evolution
2. Domains, Superfamilies, and Classification
4. Peptide Fragments
3. Nature of Fold Space & Issues with Classification
3.5 DeepUrfold

Figure 1.1: Chapter Outline.

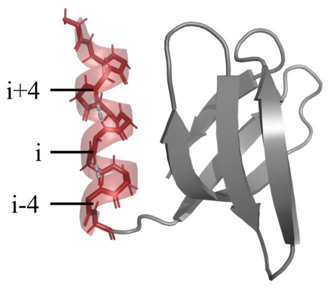
This chapter supplies background information on (1) protein structure and evolution, (2) different ways to view and organize the protein universe, and (3) several approaches to sequence and structure comparison (Fig. 1). Finally, I will present a new way to organize the protein universe, allowing for a different approach to think about relationships amongst proteins. Ultimately, the general model and approach described in this dissertation aims to develop a novel approach to find common fragments between distantly related proteins; these structural units may correspond to the precursors of modern proteins.

1.1 Background on proteins and evolution

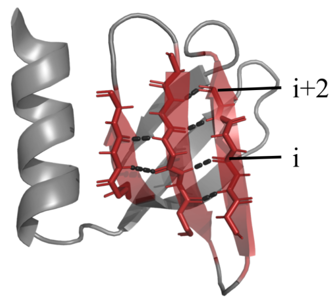
Every protein, also known as a ‘polypeptide’, is chemically built from the same 20 peptide building blocks called amino acids. Each amino acid has different geometric and biophysical properties such as hydrophobicity, steric volume, aromaticity, and polarity, giving a seemingly infinite number of ways to create different proteins via different combinations of strings of amino acids. For example, for a protein of 200 amino acids, the number of possible sequences would be 20^{200}, which is larger than the number of atoms in the universe [2]. A single protein polypeptide chain is created by translating mRNA in the protein-synthesizing machinery (the ribosome) from transcribed DNA.

1.1.1 Primary structure

The linear, one-dimensional sequence of amino acids that defines a unique protein; generally written as a string of characters from the N’- to C’-terminus [3].



(a) α**-helices** in red formed by H-bonds every fourth residue



(b) β**-sheets** in red formed by H-bonds every second residue in a single strand to adjacent strand

Figure 1.2: Secondary Structure Elements from PDB 1kq2

1.1.2 Secondary structure

Local folding of the backbone to create α-helices and β-strands, the latter of which can assemble laterally into β-sheets. Helices are formed by hydrogen bonds between backbone atoms of every fourth residue in the sequence (i to i+4) to create the helical shape. Sheets are formed via hydrogen bonds between backbone atoms, typically alternating every other residue in a strand to a neighboring strand [3].

1.1.3 Tertiary Structure

Global folding of the protein by arranging the secondary structures in 3D space. Residues that are far apart in sequence are brought into physical proximity via many factors, most notably the role of hydrophobic packing, where hydrophobic residues are packed in the center and hydrogen bonds are formed between them [3]. When we mention protein 'structures', 'folds', and 'domains,' this is generally the level to which we refer.

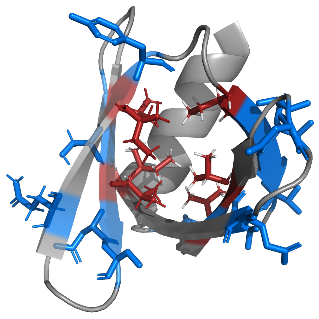


Figure 1.3: Tertiary Structure. Hydrophobic residues (red) are pushed inward, while polar residues face outward to interact with solvent (PDB ID 1KQ2)

1.1.4 Quaternary Structure

Two or more tertiary structures interacting with each other on different protein chains. Proteins of the same type can form homo-oligamers while proteins of different types for hetero-oligermers. Such associations are also called protein-protein interactions (PPIs). While this dissertation will not go into detail about PPIs, protein structure and PPI are inherently linked, and many of the same principles governing PPI also go into protein folding [4].

1.1.5 Protein Folding Problem

This is the problem of understanding the actual physicochemical process, in molecular and mechanistic detail, by which a given protein forms its 3D structure. Given that there are a colossal number of folds of a single protein could theoretically adopt, a protein cannot sample every possible fold; one hypothesis is that there are multiple routes to get to the 3D structure, often called the energy-landscape funnel [5]. Proteins are generally thought to begin forming secondary structures inside the ribosomal exit tunnel [6], but the detailed process remains unclear. There are several proposed models for small proteins on how each secondary comes together and folds on itself

[3]. While some physics-based protein structure prediction algorithms try to *learn* folding pathways, protein structure prediction is an inherently quite simpler problem, as its central challenge is to accurately map sequence $\rightarrow$ structure, without concern as to the mechanistic pathway leading to adoption of the predicted 3D structure

1.1.6 Protein Structure Prediction

This is the problem of predicting the 3D shape of a protein given only its sequence, and sequences of related proteins from the same protein family. This goal is simpler than the protein folding problem, as it neglects the biophysical mechanisms of folding. Until 2020, the most accurate structure predictors were in the form of homology modeling (such as MODELLER [7]), using known related structures from a common ancestor as a structural template, followed by threading, which uses proteins with a similar fold, but not necessarily from a common ancestor (such as I-TASSER [8]). However, if no known templates are available, these methods cannot be applied. Small proteins can be predicted from physics-based approaches, such as molecular dynamics simulations [9]. The next stage in the evolution of structure prediction approaches involved finding correlated mutations in deep multiple sequence alignments, as exemplified by the EVfold method [10]. If two residue positions mutate in similar, statistically correlated ways, then that likely reflects a pressure to maintain homeostasis and keep the hydrophobic core intact; this, in turn, implies that the positions are likely to be close—or at least somehow physicochemically *coupled*—in 3D space. The most recent and highly successful structure prediction approach is AlphaFold2, which predicts structures using deep learning [11]. EVfold and AlphaFold2 are further discussed in the protein comparison section.

1.1.7 Molecular Evolution at 1000 feet

All of the different shapes and functions proteins can perform are the result of over 3.8 billion years of evolution. While there are many open questions regarding molecular evolution, some of which will be described in the subsequent chapters, a few fundamental concepts will be explained here. Each protein is coded for by a gene in the DNA sequence. Slight changes to the genomic DNA, due to insertions, deletions, and mutations from an error in the translation machinery, radiation, or exposure to chemicals (mutagens), can have a downstream impact on a protein's 3D shape and function. These mutations can be 'silent' (unchanged amino acid), 'nonsense' (terminates the protein prematurely), or 'missense' (the amino acid is changed to a new one from the original, 'wild-type' sequence) [12].

The changes in protein sequence may be neutral, thus propagate through a population randomly via a stochastic process known as 'neutral drift', or can be actively selected for or against. If the change increases the organism's likelihood of reproducing, it can be viewed as beneficial and likely will be maintained in the species ('positive' or 'balancing' selection) [13]. If the change is deleterious ('negative selection'), it will be removed from the species lineage. Different versions of a gene can occur when the gene is duplicated in the genome (paralogs), which can change the 3D shape and possibly confer new functionality to the protein (neofunctionalization). Indeed, gene duplication followed by drift and diversification is hypothesized to be one of the most important means of molecular evolution. The basic rationale is that it is simpler to re-use existing parts in order to elaborate new functions rather than create new parts and functions *de novo* [1].

1.2 Domains as the unit of evolution

Each protein can have, depending on its size, one or more independently folding units, which we call a 'domain' [14, 15]. The protein domain is traditionally considered to be the unit of function and evolution. Domains from different proteins that share similar sequence and structure usually perform at least roughly similar functions: indeed, this is molecular biology's foundational sequence/structure/function paradigm. In terms of evolution, new functions can emerge for a single domain (not the entire protein), and various domains can be duplicated and mixed to form larger, multi-domains proteins; entire domains can be copied into new hosts (and species) via horizontal gene transfer. All of the evolutionary aspects described above could occur at the domain level rather than the whole protein; in this way, domains can be viewed as modular units of sequence/structure/function. For all these reasons, organizing, clustering, classifying, and labeling at the domain level has been the primary approach to understanding the protein universe.

Hierarchical classification schemes based on 3D structure have become the most popular approach to identify domains that share similar functions, shapes, and evolutionary histories. The most common resources for protein structure organization are three independent databases: the Class, Architecture, Topology, Homologous Superfamily database (CATH; [16]); the Structural Classification of Proteins (SCOP; [17, 18]) database; and the Evolutionary Classification of Domains (ECOD; [19]) database. At the sequence level, there also exist resources of similar domains, such as Pfam [20]; however, since they are purely sequenced-based, we will not cover them in this section. Note that each of these domain-based resources have different definitions of strict domain boundaries—i.e., where a domain starts and ends in the sequence (and structure), due to varying thresholds of inter- and intra-domain contact densities. A 'domain' to an evolutionary biologist might differ significantly for a structural biologist or a systems biologist, where a domain could be a recurring unit in many proteins, have a conserved core, or be functionally similar, respectively [21]. Understanding differences between structural hierarchies such as CATH, SCOP, and ECOD will give insights into multiple views of the protein universe.

The most widely used classification scheme is called CATH [22], which is based on four main hierarchical levels:

1. **Class:** predominant type of secondary structural element (SSE) content (all-α, all-β, mixed α/β, and small proteins);
2. **Architecture:** global arrangement (relative positions and orientations) of the main SSEs in 3D space;
3. **Topology:** the pattern of connectivities (one-dimensional orderings) of the SSEs; and
4. **Homologous Superfamily:** a collection of domains that share at least 20% sequence identity and are thought to have a common ancestor [22].

The CATH database has seen massive growth going from 53 million protein domains classified into 2737 homologous superfamilies in 2016 to a current 95 million protein domains classified into 6,119 superfamilies.

SCOP [17] organizes domains across the following four hierarchical levels:

1. **Structural Class:** types of secondary structure elements (all-α, all-β, α/β, $\alpha+\beta$, and small proteins); SCOP further separates 'α/β' into 'α/β,' and '$\alpha+\beta$'

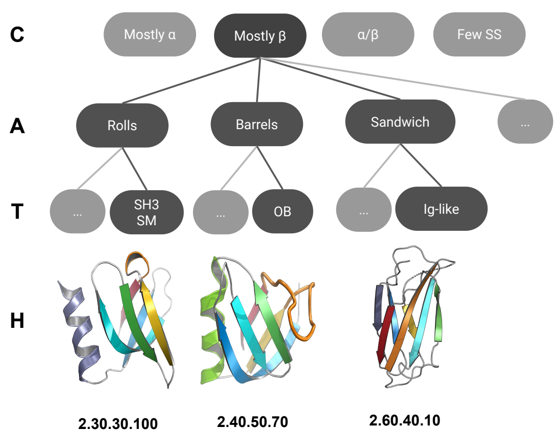


Figure 1.4: CATH Hierarchy highlighting the OB and SH3 relationships.

to represent the difference between proteins alternating α helices and β strands and those with all α helices separated from all-β strands.

2. **Fold:** Groups of 3D structures that have similar global layout (architecture) and topology of SSEs; note that this level corresponds most closely to the 'T' level in CATH.

3. **Superfamily:** Groups of distantly related domains with conserved 3D structures. These domains may have a common ancestor, but it's not required.

4. **Family:** Groups of domains with a clear evolutionary origin.

There are currently 5,860 Families, 2,785 Superfamilies, 1,550 Folds, and 5 classes in SCOP.

ECOD [19] is the most recent of the three protein classification databases, and it organizes each domain using five hierarchical levels:

1. **Architecture:** Groups of domains with shared SSEs in specific 3D shapes and locations.

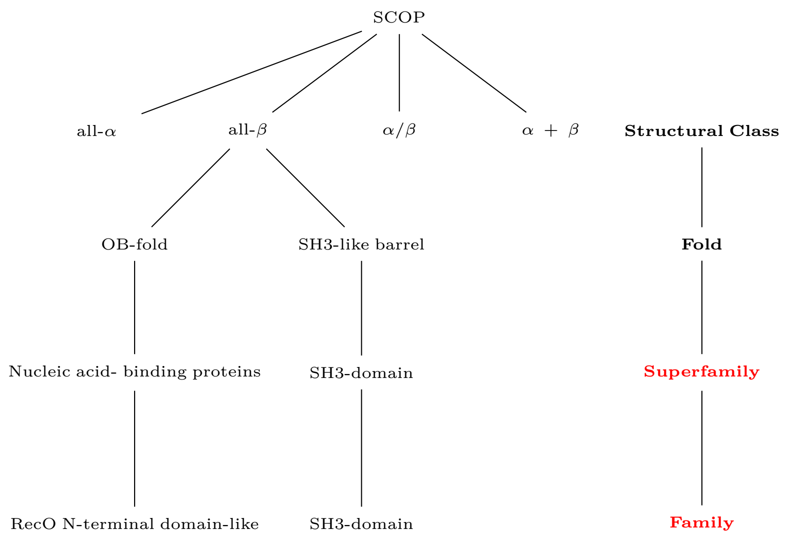


Figure 1.5: SCOP Hierarchy highlighting the OB and SH3 relationships. Superfamily and Family levels (red) are based on evolutionary relationships, while Structural Class and Fold are not.

2. **X-group:** Groups of domains that share structural similarity and may be homologous albeit without strong evidence.

3. **H-group:** Groups of domains that are homologous based on sequence- and structure-alignment, functions, and literature searches.

4. **T-group:** Groups of homologous domains with similar connection between their secondary structure, e.g. topology.

5. **F-group:** Groups of homologous domains with the same topology with significant sequence similarity.

ECOD has 16,176 F-groups, 3,715 H-Groups, and 2,460 X-Groups [19]. Each of these hierarchies differ from one another in systematic ways. Not only do the predicted domain boundaries differ between the different methods, the underlying algorithms of each method are quite different as well, each with their own strengths. ECOD and

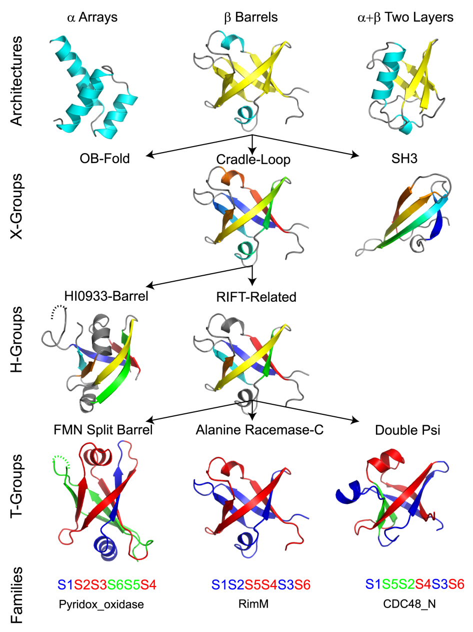


Figure 1.6: ECOD Hierarchy highlighting the OB and SH3 relationships. Copied from [19] figure 2, which is under the CC-by 4.0 license.

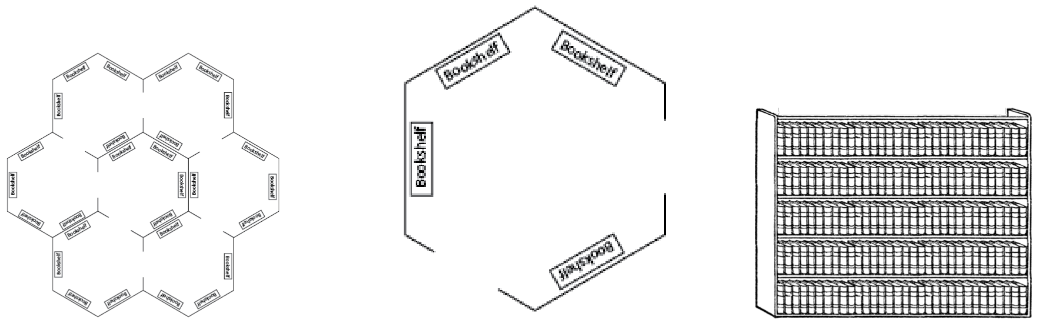
SCOP have the most similar domain boundaries, which are different from CATH. In a recent comparison of the three hierarchical classification schemes, it was found that CATH tends to favor more compact domains, while ECOD enriches for distant evolutionary relationships and SCOP performs well at identifying non-redundant representatives [23].

1.3 Nature of the protein universe and issues with hierarchical classification: discrete vs continuous

Despite being perhaps the most comprehensive resources available, CATH, SCOP, and ECOD have intrinsic limitations in their design and structuring–reflecting assumptions and constraints that are inherent to any hierarchical classification system. Their specific hierarchical schemes implicitly (and strongly) influence how we perceive the protein universe, or protein fold space. The protein structure universe is the collection of all possible structures that proteins can adopt. This includes all possible mutations and fold variants that have not been thought of yet, or sampled during evolution. The organization of this fold space could give clues about protein interrelationships and protein evolution. Understanding which regions in fold space are populated and how densely those regions are packed, is likely to implicitly harbor deep information about protein interrelationships over a vast multitude of protein evolutionary timescales, ranging from ancient, primordial motifs that have persisted in modern proteins, to more recently-arisen folds and structures.

When dealing only with protein sequences, we call it. ‘protein sequence space,’ which is a somewhat more intuitive concept than fold space. Protein sequence space is the collection of all possible protein sequences, that is to say every possible ordering of n amino acids for a protein of length n residues. Assuming all proteins are 200 amino acids long, protein sequence space would have 20^{200} possible sequences – a seemingly infinite amount. One way to comprehend these seemingly infinite possibilities has been explored by Jorge Luis Borges, an Argentinian author, in his short story ‘The Library of Babel’ [24]. Borges describes a library containing every possible book that has or ever will be written, each 410 pages long with a standard alphabet in the same format. As you can imagine, most of the books are filled with nonsense, but others may have intelligible stories. For example, this completely written thesis is already in the library even if it is not finished yet. Travelers spend their entire lives exploring the library in the hopes of finding meaningful phrases.

The Library of Babel is a sphere and ‘organized’ into hexagonal rooms with 4 walls of



(a) Entire Library **(b)** Single Room **(c)** Single Bookshelf

Figure 1.7: Hierarchical Organization of The Library of Babel. A) The entire library of every possible book ever written is a representation of a perhaps infinite 'space' or 'universe.' This would be equivalent to all of fold space. If viewing the fold space from the CATH lens, each hexagon could a different class. B) Each hexagonal room represents an entity in the first level of the hierarchy, in this case a single class such as 'all β.' C) One wall of the hexagon is the next level of the hierarchy, which would represent 'architecture.' The third level of the hierarchy is shelf, which could be 'topology,' each book could represent 'superfamily,' and each page could represent a single domain in the superfamily. Bookshelf image from (C) are copied with permission, courtesy of Jonathan Basile via his site `https://libraryofbabel.info/`, which is under a Creative Commons license.

books, 5 shelves per wall and 32 books per shelf. The two remaining walls are doors to other rooms. The hexagonal organization is similar to the hierarchical nature of fold space, yet is still meaningless. Imagining a different organizational scheme would allow library goers—travelers—to find their books more easily.

Most of the library is filled with nonsensical books, which can be seen as protein sequence space containing nonsensical proteins that will not fold. As a protein's sequence and its structure are inherently linked—Anfinsen first showed that a protein's 1D sequence encodes its ability to spontaneous fold to a native 3D structure [25]—there exists a relationship or 'mapping' between protein sequence space and protein fold space; that complicated relationship has not been fully explored and is beyond the scope of this dissertation.

Before proceeding, we must define what a mathematical 'space' or 'universe' means. This is important conceptually (for logical clarity and consistency), as well as pragmatically (for precisely defining what we mean by computed properties, statistics, etc.). There are many types of mathematical spaces, and the ones that are most relevant here are 'Euclidean space', 'metric space' and 'vector space'. In Euclidean space, there exists a single origin and the positions of each object (in this case, a single protein) can be described by a vector; in 3D, this could be the (x, y, z) coordinates of each entity comprising the object (e.g., atoms in a protein, or entire proteins in a 'protein structure space'). In a metric space, all objects in the set have a 'distance' or 'similarity' function that can be computed to give a distance between them; however, that function does not necessarily have to be the straight line (Euclidean), and the metric space does not necessarily have a zero (the origin). In a vector space, there is a clear origin, all objects are defined by a vector from the origin, and linear combinations of vectors allow converting objects into other objects.

Do individual proteins populate a generalized protein structure space in a discrete or (essentially) continuous manner? At this time, we see protein domain structures from different non-extinct organisms with the same fold all with different functions per fold, that seem to cluster in their own independent 'islands'—i.e., a discrete organization, which might be amenable to hierarchical classification. When we think of these different islands, it implies that protein fold space is discrete, wherein we cannot traverse space island to island, or cluster to cluster, because the folds are so different and there are no well-defined or discernable relationships between them. Because there exist quantities that can be readily computed (e.g., the RMSD, see section on protein structure comparison) in order to 'measure' the distance or similarity between protein domains, people usually think of fold space as a metric space; however, the similarity metric becomes ambiguous for distantly related proteins having random similarity. By binning proteins into discrete, mutually exclusive, categories, CATH, SCOP, and ECOD all assume a discrete underlying representation of protein fold space. However, viewing fold space as purely discrete may lead us to miss remote connections and disregard the role of evolution. An open question in biology concerns the mechanisms by which domain structures arose—it is unlikely that folds arose/evolved independently, suggesting a more continuous nature of fold space, wherein evolutionary transitions can occur between folds A $\rightarrow$ C via some intermediate, say B. In this sense, the discrete versus continuous duality of protein fold space can be viewed largely as semantics–i.e., a matter of thresholding [26].

An alternate view of protein fold space regards transitions within it as a continuous process. All protein structures are related through evolution, so it should be possible to traverse fold space by combining short secondary structure segments, or mutating a structure to get to different possible folds, akin to a 'vector space.' Network representations have been a powerful way to describe a continuous fold space. Each node in the network is a domain and edges are drawn between them if they have a similarity score larger than a given threshold, usually determined via all vs. all domain structure alignment. These networks are nearly connected and each domain can be traversed to and from different domains in around 4-8 steps [27]. It has been noted that many of the folds that are connected share similar small peptide fragments, which will be discussed in the next section [28, 29, 30]. Another method to visualize a continuous fold space is to convert the similarity scores from all vs. all domain structure alignments into pairwise similarity matrices where each domain has attributes of scores to every other domain. The matrix can be reduced to two or three dimensions using Principal Components Analysis (PCA) or Multidimensional Scaling

(MDS) in order to visualize the most significant degrees of freedom. The resulting graphs, shown in Figure 8, display protein domains in the 'all-α' and 'all-β' classes as being furthest apart, separated by protein domains in the 'α/β' class. Protein domains in the 'α/β' class form a diverse core, and the further away from the core, structures become less diverse [31, 32].

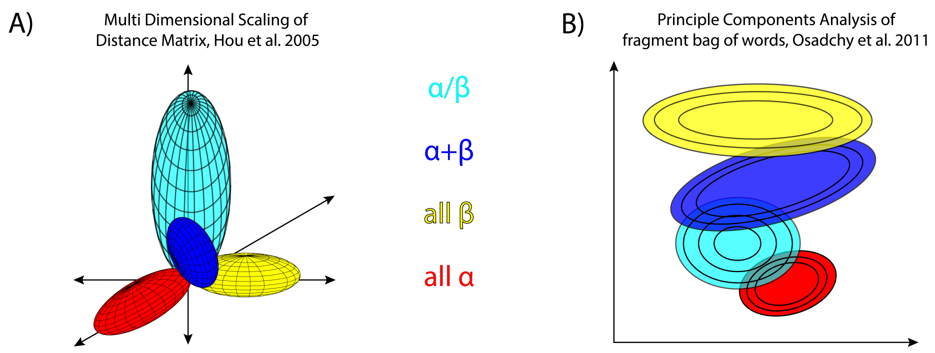

A)
Multi Dimensional Scaling of
Distance Matrix, Hou et al. 2005
α/β
α+β
all β
all α
B)
Principle Components Analysis of
fragment bag of words, Osadchy et al. 2011

Figure 1.8: Different Views of Continuous Fold Space showing protein domains from 'All α' and 'All β' classes flanked by protein domains from the 'α/β' classes from different comparison metrics six years apart. A) Each distribution consists of points that represent one protein and a point's feature vector contains the distance to every other protein in the dataset (a distance matrix), clustered with Multi Dimensional Scaling (Recreated from a similar image from [31]). B) Each distribution consists of points that represent one protein and a point's feature vector is the presence of specific fragments from FragBag (a bag of words representation), clustered using PCA (Recreated from a similar image from [32]).

The discrete vs continuous duality of fold space has been an unresolved question in molecular evolution and structural bioinformatics. It has also been observed that protein space is more discrete at high similarity thresholds and continuous at lower similarity thresholds, which does not answer the question and leaves many questions about how proteins evolve [26, 27].

1.4 Short peptide fragments as unit of evolution

In light of these issues described above, an alternate way to describe proteins is through short contiguous peptide fragments that make up each domain, rather than viewing the full domain itself as the elementary/modular unit of evolution. Presumably, the protein universe did not spontaneously arise with intact, full-sized domains from the start, so there must have been a precursor to a complete protein domain [1, 33, 34]. From the earliest days of protein structure determination and analysis, common structural motifs have been noted and investigated [35, 36]. Small fragments were first used to understand the protein folding problem – not for general evolutionary analysis. While there have been many approaches to investigate the use of short peptide fragments, I will focus on a few that I think are most relevant.

The first wave of automated fragment identification methods involved splitting proteins into short segments of 4-10 amino acids and clustering the fragments into smaller sets using unsupervised machine learning methods [39, 45, 37]. The next wave of small fragments came about while studying all vs all relationships of the available protein structures at the full domain level. In a landmark study, Holm & Sander [28] created an all-by-all similarity matrix from structural alignments, reducing its dimensionality with multi dimensional scaling. In this new 2D similarity space, they found 5 'attractors' or short peptide fragments common to all of the structures [28]. In another study [55], Harrison et al. developed a new similarity metric through a fully connected graph representation of secondary structures. Each node was a single SSE that was connected by an edge to every other SSE. Edges were weighted by distance, angle, dihedral angle, and chirality. To compare two proteins, they created a product graph of both protein's graphs and found maximal common cliques taking into ac-

Name	Data source	Contiguous	Fragment length	Parser	Library	Search	Evolution
Building Blocks (1989) [37]	Struc	x	6	x	60		
Motifs (1995) [38]	Seq	x	-	x	3400	x	
Patterns (1996) [39]	Seq	x	3-15	x	13,151	x	
Attractors (1996) [28]	Struct	x			16 / 5		x
Spacial Motifs (1999) [40]	Struct	x		x	x	x	
Common Substructures (2000) [41]	Struct		75-281	x	>75		
Protein Blocks (2001) [42, 43]	Struct	x	5	x	12	x	
FragBag (2002) [44, 45, 32]	Struct	x	5-12	x	x	x	x
Fragnostic (2005) [46]	Seq	x	5-20	x	x		x
Structural footprints (2007) [47]	Struct	x	32	x	x	x	-
SISYPHUS (2007) [48]	Struct	x	20–50		x	x	
Smotifs (2010) [49]	Struct	x	2 SSE	x	324		
Elementary Functional Loops (2010) [50]	Seq	x	6-41	x	37		x
Primordial Peptides (2015) [1]	Seq	x	9-38	x	20		x
Themes (2017) [29, 34]	Seq	x	35-65	x	2195	x	x
DeepFold (2018) [51]	Struct	x	-	x	400	x	
Protodomains (2019) [52]	Struct	x		manual		manual	-
Structural Motifs (2020) [53]	Struct	-		x		x	
Shapemers (Geomtricus; 2020) [54]	Struct		10-15	x		x	x
Structural Motifs (Sahle; 2022) [33]	Struct	x	7-68	x	6	x	x

Table 1.1: A historical perspective on short peptide fragments. Contiguous=The algorithm requires fragments to be adjacent in the sequence. Parser=Describes how fragments were identified. Library=If the method produces a library of fragments (clustered and/or unclustered), the number of fragments in the library is shown. Search=the paper used those fragments to identify other proteins in the PDB. Evolution=The paper took an evolutionary perspective, rather than being limited to just structure analysis,prediction or engineering/design.

count the edge weights. They found that 80% of folds shared common cliques, which represent short structural fragments, with other folds, which they call 'gregarious' [55]. These two approaches pushed the thinking that short fragments are possible units of evolution as well as general building blocks for full protein domains.

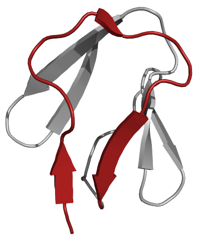
An algorithm known as 'Fragnostic' was one of the first to treat structure space as a graph of domains connected by an edge if they shared any fragment, rather than if above a certain threshold of a similarity score derived from a full structure alignment [46]. They found a few folds with many connections and lots of folds without many connections, consistent with a power law distribution [46].

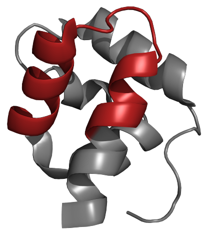
The next generation of approaches were to describe a single protein by a vector of fragments. In [47], Structural footprints identified short fragments using 14 shape descriptors based on a writhing number—i.e., how often two SSEs cross each other's path in 3D–which are then clustered using k-means. Structural similarity is calculated by taking the Pearson correlation of the fragments. Next came FragBag [44], where fragments from [45] were clustered to form a new smaller library. A protein could then be described as a feature vector that is the length of the new smaller library, with vector elements being the number of occurrences of each fragment. FragBag is able to identify structural neighbors faster than doing all vs all structural comparisons [44].

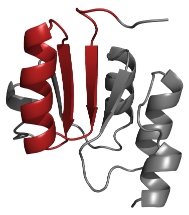
Many databases of short peptide fragments followed. SISYPHUS was developed using methods described above [48]. Another database called Smotif [49] was developed using two consecutive SSE to find fragments, which were then used to search the PDB to find novel folds. While all of the methods described above were developed for proteins in general, the next wave of fragment identification approaches came about while studying specific classes of proteins, such as enzymes and metal-binding proteins. Evolutionary implications of short peptide fragments became more apparent after combining the short fragments with sequence comparison tools that use statistical frameworks for protein evolution. In [50], Goncearenco & Berezovsky identified common loop fragments flanked by SSEs, or Elementary Functional Loops (EFL), that join together in 3D space to become specialized to perform enzymatic activity. EFLs are found by tiling a set of non-redundant sequences into overlapping fragments and searching a library of proteins for each fragment iteratively using position-specific scoring matrices (PSSM) to create profiles for each fragment, similar to PSI-BLAST [56] (see protein comparison section). Profiles below an E-value of 1 were kept as potential EFLs.

Some methods use an altogether different way to identify common fragments across different proteins. Instead of using pre-cut fragments, [1] and [34] start with a set of non-redundant complete protein domains, from SCOP and ECOD respectively. A profile is created for each protein using HHsearch. In [1], Alva et al. saved matches from HHSearch if the match had a probability $\geq 70\%$, a structural alignment score (TM-score) of ≥ 0.5 (see protein structure comparison section), and were assigned to a different SCOP fold. A set of 40 fragments were kept by finding matches that have $\geq 80\%$ overlapping regions of number residues, clustering, and were present in the Smotif dataset. Through duplication, recombination, and drift, they hypothesize that these short peptide fragments, called 'primordial peptides,' combined to create all known protein structures. These fragments have all been found bound to RNA, suggesting they could have arisen during the RNA world. However, they found no domains that had two different primordial peptides [1].

In [34], Nepomnyachiy et al. defined a new scoring function to save a match from HHsearch based on BLOSUM62 scores of the alignment. A dynamic programming al-


(a) β-Hammerhead. Primordial Peptide 12 (d2je6i2).


(b) Helix-Hairpin-Helix. Primordial Peptide 2 (d1ixra1).


(c) β-α-β. Theme 2939 (e4e4tB7).

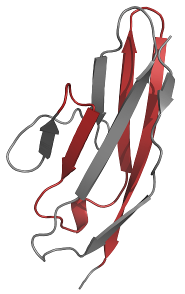

(d) β-β-β. Theme 16416 (e2v5A1).

Figure 1.9: Example short peptide fragments. From 'themes' [34] and 'ancestral peptides' [1].

gorithm was used to find an optimal set of non-overlapping fragments, called 'themes', with a maximal score using different fragment lengths. Two themes from different folds are considered the same if they have $\geq$ 85% sequence similarity, and similar themes are clustered based on a structural alignment distance (RMSD, explained below in the structure comparison section) to obtain 2195 total themes [34]. The sizes of libraries created by these approaches [1, 34] vary greatly, reflecting different stringencies of thresholds and ultimately different goals. However, both the 'primordial peptides' in [1] and the 'themes' in [34] are currently the state-of-the-art fragment libraries with examples shown in Fig 1.9.

Another recent approach of finding short peptide fragments has been through visual inspection of symmetry in protein domains. According to [52], 'proto-domains' often assemble with the same proto-domain in C2 symmetry. Youkharibache's method is completely manual, but offers important insights on why these peptide fragments might associate to form complete domains. Self-assembly, often via a symmetric organization, is a key physical driving force in evolving and maintaining domains, and is especially notable at the quaternary structural level. However, just as with Alva et al [1], no domain was found to have two or more different proto-domains [52]. Youkharibache's hypothesis is compelling, particularly at explaining why certain fragments are retained throughout evolution (versus the evolutionary mechanism itself).

All of the previously described methods make the assumption that these short peptide fragments are contiguous in sequence. One of the more recent approaches, Geometricus [54], can find peptide fragments that are not contiguous using moment invariants on residues that are within a given radius in 3D space. Contiguous fragments are also obtained and combined with the radius approach to create fragment embeddings for each protein of interest. Discontiguous protein fragments that are shared between proteins may be the next piece to create connected linkages through more of fold space.

1.5 Protein Comparison

In all of the methods described above to understand fold space, similarity measurements had to be defined between proteins. Ideally, this metric would identify a common ancestry and accurately quantify how much two proteins diverged if they are homologous. A natural approach would be to compare sequences, typically through an alignment. Sequence alignments work well especially due to a strong statistical framework. If a sequence has significant non-random similarity, it is likely that the two proteins are homologous. However, these approaches cannot detect distantly related proteins, below a so-called ‘twilight zone’ of similarity (20-30% pairwise sequence similarity) [57, 58].

Structure similarity through a structural alignment can be used to find relationships in the twilight zone of sequence similarity because 3D structure is typically more deeply conserved than is sequence. Determining homology via structure, however, is much more difficult because there is not a strong statistical theory—two structures could randomly be similar or the similarity may have been the result of convergent evolution; there is no probabilistic model to supply a ‘null hypothesis’ for structure comparison, unlike, for instance, a well-defined random sequence model (which is readily constructed and compared against). In this section, I will give a brief historical note about sequence and structure comparison ending with current best practices in the field.

1.5.1 Sequence Comparison

One of the first methods to compare protein sequences did so via ‘global alignment.’ Two protein sequences were aligned from N’ to C’-termini, finding the optimal match between each residue allowing for matches and insertions/deletions (i.e. ‘indels’, accounted for via ‘gaps’ in the alignment). The Needleman-Wunsch (N-W) algorithm [59] was developed to find the optimal alignment through dynamic programming by scoring every possible alignment as a matrix of the two sequences being compared. Each cell in the matrix contains the score for a match, insertion, or deletion in relation to the previous cells, starting with simple scores such as match=1, insertion=deletion=-1. Once the scores have been calculated for each cell, an optimal path backwards can be obtained by finding the max scores from the lower-right to upper-left via a traceback procedure [59].

However, global alignment does not always produce adequate alignments for distantly related proteins, and local segments that were more similar (than their flanking regions) were often missed by the algorithm. The Smith-Waterman (S-W) algorithm [60] was developed to find similar local regions of sequence via a 'local alignment.' Using a similar dynamic programming algorithm to N-W, they set negative scores to zero, start their traceback procedure at the highest score, and add score penalties for expanding gaps [60].

Both the N-W and S-W algorithms were further improved by the addition of substitution matrices accounting for the probabilities of changing from one amino acid to another based on alignments for conserved protein families. The most notable similarity matrices (often called 'scoring matrices') were the Dahoff frequency tables (Percent Accepted Mutations; PAM) or BLOSUM. Dayhoff matrices were constructed using the observed frequencies of each substitution where each substitution is independent in sequences with greater or equal to 85% sequence identity. BLOSUM matrices clustered proteins at different similarity thresholds (most common being BLOSUM62 for 62% sequence identity) using a log-odds probability for observed mutations [61].

The methods described above were still too slow to search large databases, so the next improvements to sequence alignments were the use of k-mers, short stretches of *k* amino acids. The FASTA algorithm was one of the first methods to *k*-merize the sequence [62]. The next improvements came by removing low-complexity regions to increase speed and applying a solid statistical theory to predict the probability of seeing a sequence by chance in a database of a certain size, implemented as the well-known Basic Local Alignment Search Tool or BLAST [63, 56]. BLAST was further improved by adding multiple iterations, while updating a PSSM [56]. Further speedups can be attained, for example, by pre-filtering large databases using a reverse k-mer index (MMSeqs2 [64]).

Hidden Markov models (HMMs) were the next major breakthrough in the field of sequence alignment and comparison. HMMs are probabilistic graphical models with transition probabilities from one state to another: various amino acids can be viewed as being 'emitted' (probabilistically) from the discrete 'states' of the HMM, and transitions from state $i \rightarrow i{+}1$ thus build-up a one-dimensional string of characters (i.e., the protein sequence). The first HMMs in biology modeled sequences as transmissions between states corresponding to matches, insertions, and deletions. An HMM can then generate new sequences by 'walking' through the model by following the trajectory of probabilistically-determined state transitions. The first approach was to use known alignments to train each model and learn the transition probabilities. An alternate approach is to learn the transition probabilities from unaligned sequences using the maximum expectation algorithm to maximize the log likelihood each sequence came from the model [65, 66]. Once you have a trained model, you can obtain the probability any sequence arose from the model, resulting in fast database searches. Common HMM implementations in biology are Sequence Alignment and Modelling (SAM [67]) and HMMER [68] to create a 'profile' of a given multiple sequence alignment. Improvements include HMM to HMM comparisons (e.g. in HH-suite [69]) and iterative approaches of creating a model (profile) for a single sequence, searching for other sequences in a database that fit the model, adjusting the transition probabilities in the position-specific profile, and searching the database again with the updated model (e.g. jackhmmer [70]). Iterative approaches are now the de facto method to build large, diverse, deep multiple sequence alignments (MSA) used to train newer machine learning models. Most HMMs are built for single sequence families, from prebuilt multiple sequence alignments, often structure-based alignments.

The next wave of sequence comparison methods came in the form of Potts models. Having originated as generalized Ising (lattice) models in statistical mechanics, Potts models can also be derived from the maximum-entropy principle and are used to find global correlations (hence their utility in protein sequence comparison). Potts models are used to find correlated mutations in the columns of a deep MSA of a protein family, often used to predict the structure of a single protein from the same protein family [10, 71]. These models were then used to compare a sequence from the same family (in the same MSA) to understand how well they tolerate mutations [72].

Finally, Deep Learning (DL) and Natural Language Processing (NLP) have been the latest machine learning approaches to be used for sequence comparison and analysis [73]. With extremely large databases of protein sequences (e.g. Uniprot [74] and BDF [75, 76]) and hardware advancements (GPU computing), deep learning on proteins has now become extremely popular. The first approaches of DL and biology were for function prediction using supervised learning on known positive examples, where comparisons were used to identify similar functions [73]. One important aspect of the new DL methods is that comparisons do not require alignments, also known as 'alignment-free.'

DL architectures for protein sequence began by chunking the sequence into many fragments; in Deep Learning, this approach is considered a Convolutional Neural Networks (CNN) in one dimension. However, CNNs were not as good at capturing long range dependencies. More advanced models now take into account the previous position in sequence, rather than tiling into fragments, e.g. in Recurrent Neural

Networks (RNN). An example of an RNN is the Long Short-Term Memory (LSTM). Bi-directional LSTMs were invented to account for next and previous positions in a sequence [77]. Another approach has been using autoregressive models [78], where each position takes in all of the preceding positions as opposed to Potts-based model, EVcouplings and DCA, which only compare pairwise columns of a multiple sequence alignment [79, 80]. Masked language models have also been used to delete amino acids at certain positions and have the model correct the missing residues. The most recent wave of DL approaches has involved “attention,” where all positions in a given sequence can ‘attend to’ every other position in the sequence, allowing the model to learn which residues are most important/informative [77].

One of the most important deep learning methods has been the Autoencoder, which compresses an input, such as a protein sequence, into new, lower-dimensional representation, also called an embedding, vector known, or latent space, which is then used to reconstruct the original input. Later, autoencoders were used for variational inference, in a model called the Variational Autoencoder (VAE), that learns the distribution of the input, eg. the mean and variance of a normal distribution as its embeddings. VAEs then sample from that distribution to create a similar, yet different, embedding, which is then reconstructed to resemble the original input. The mean square error of the actual input versus the reconstructed version, with the entropy between learned and true distributions, called Evidence-lower Bound (ELBO), are used as the VAE loss function to improve the learned distribution [79]; this topic will be discussed more thoroughly in Chapter 3. VAEs are considered a ‘deep generative model’ because they are able to generate ‘new’ or unseen examples that are similar to the input.

The first studies for DL methods were superfamily specific and compared differences between sequences in the same superfamily. DeepSequence used VAEs and compared ELBO scores to see how well aligned sequences in the same superfamily tolerated mutations [79]. SeqDesign uses cross entropy loss to compare (unaligned) sequences in the same superfamily using autoregressive models [78].

Next, larger language models were built using larger sequence databases such as Uniprot [74] and BFD [75, 76], allowing for the comparison across the entire sequence space, removing the need to only compare proteins in the same family. The output of these language models, called an embedding or latent space, is what is used to compare to proteins. The Manhattan distance or Euclidean distance between the embeddings measures semantic similarity. The language models have been used for contact prediction, sequence alignment, variant prediction, and general function prediction. Examples of current language language models include UniRep [81], TAPE [82], ProtTrans [83], ProGen [84], ESM [85], and MT-LSTM [77]. Language models have also been used for sequence alignment and deep homology searching [86, 87, 88, 89]

1.5.2 Structure Comparison

Structure comparison approaches have lagged behind sequence comparison methods. This is partly because it is much more difficult to make claims of homology using structure–at least with well-principled statistical confidence bounds–as there is no statistical theory that is as well-grounded as for sequence comparison. The first approach to compare structures (dating to the 1970-80s) was via structural alignment, or 'superimposition,' using rigid-body least-squares fitting, e.g. via the classic Kabsch algorithm to compute the rotation matrix that minimizes the RMSD between the 3D coordinate sets of two proteins [90]; an equivalent (yet more general) way of viewing the problem is as a singular value decomposition on a cross-covariance matrix of both sets of atomic coordinates [91]. However, while this produces meaningful results for nearly identical proteins; proteins with many insertions and deletions caused many problems, motivating the next generation of structure alignment algorithms.

The next generation of structure aligners, pioneered largely in the 1990s, focused on aligning smaller fragments, usually single secondary structures or super secondary structure elements, and optimizing the alignments based on different distant measurements, often using dynamic programming or Monte Carlo-based approaches. Examples of this family of methods include Sequential Structure Alignment Program (SSAP [92]), Distance-matrix Alignment method (DALI [93]), Combinatorial Extension (CE [94]), and Vector Alignment Search Tool (VAST [95]).

All of these methods rely on relatively crude definitions or measurements of pairwise 3-D 'similarity'. The most common distant measurement is the Root Mean Square Deviation (RMSD) of the aligned C_α atoms, where two identical point-sets give a value of 0 and anything larger than that means greater dissimilarity. Dissimilar proteins without many aligned residues are usually assigned an RMSD of 1000 and outliers highly influence the score. Another similarity gauge is the Global Distance Test (GD-TS) [96], which compares the number of C_α atoms aligned at different thresholds (0.5-10 , with 20 steps, and the GD score is averaged for each step). The best possible GDT_TS is 100 and the worst is 0. A GDT_TS score ≥ 50 represents a similar architecture, a GDT_TS ≥ 70 represents a similar topology (roughly, 'fold'), and two proteins with a GDT_TS score ≥ 80 likely belong in the same family.

Improving distance scoring functions to create a more robust statistical theory was the next step in improving structure alignment. One of the most common aligners, TM-Align [97], uses the Template Modeling score (TM-score [98]); this method produces alignments of higher quality because it normalizes alignment errors using a length-dependent scale, ignoring bias of random similarity [98].

Most of the recent developments in protein structure alignment have focused on creating all-vs-all pairwise structural alignments and multiple protein structure alignments, exemplified by programs such as mTM-align, MATT, and foldseek [99, 100, 101, 102, 103]. A recent algorithm, Caretta [104], takes a different approach to alignment. Caretta uses the Geometricus [54] algorithm to fragment the structure using different moment invariants, similar to the FragBag [44] approach. Aligning similar fragments computed across different proteins results in a multiple structure alignment.

Deep learning approaches have also transformed 3D structure comparison [105, 106]. 3D Convolutional Neural Networks (3DCNNs) were first applied to 3D structure for enzyme prediction [107], ligand binding [108, 109], and interaction prediction [110, 111, 112]. Protein structures are converted to a 3D volume by discretizing the Cartesian coordinates into 3D volume elements (voxels). Graph-based approaches have also been widely used, where atoms or residues are nodes that are joined by an edge if they are nearby (≤ 5) in 3D space [113, 114, 115]. These methods employ fundamentally different types of representations, or encodings, of a 3D structure. Equivariant graph neural networks [116, 117] have been addressed to fill the gaps between 3D CNNs and graph based approaches. However, most of these methods are for function prediction/annotation or discrete structural class prediction in general, and not necessarily about directly comparing two proteins.

AlphaFold2 [11] was the first structure prediction method to achieve a GDT_TS score > 80, indicating it can map protein sequences $\rightarrow$ structures with family-level accuracy. The approach they used combined sequence-based approaches (such as described above) to create deep MSAs for the family that structure of interest is from, and then used 3D Equivariant Neural Networks to predict the final structure [11]. While AlphaFold2 has been used thus far only for structure prediction, new studies of the latent space it produces should give clues to protein structure evolution and comparison methods to understand how two proteins are related.

Structure comparison and alignment remain difficult. There is still room for improvement. For example, most comparison tools still require alignments and still operate in 3D ('real') space. Also, most existing methods are topology-dependent, by which we mean that architecturally similar proteins—even at the level of identical folds (were we to neglect the pattern of links between SSEs)—would not be detected as such. ML-based methods are amenable to being naturally topology-independent because they use different representations of 3D structure, free of the constraints of identical connection patterns between SSEs. It makes more sense now to use compressed structure representations from an embedding or latent space. Current aligners also usually only take into account raw Cartesian coordinates. Because protein functionality evolves not solely dependent on atomic coordinates/geometry alone, we believe it more optimal to include biophysical and evolutionary information directly into the model.

1.6 Overall Problem

If fold space is indeed a continuous vector space, there are likely to be many remote connections between different superfamilies that are currently missed in current hierarchical representations of fold space. Importantly, we emphasize that this limitation is intrinsic to the hierarchical nature of modern classification schemes: a specific protein is discretely binned into one of several mutually exclusive categories (at all granularities of representation levels, from class to architecture on down), and doing so then precludes possible (remote) connections from being captured and represented. In addition to small peptide fragments bridging fold space, the Bourne lab recently proposed an ‘Urfold’ model to bridge the gaps in our view/understanding of fold space, by allowing for the recognition of connections between superfamilies that have the same 3D architecture but with permuted SSEs [118, 119]. If viewing fold space through the CATH lens, it might represent an intermediate level of structural granularity that lies between the CATH hierarchy’s architecture (A) and topology (T) strata. The Urfold representational level is thought to capture the phenomenon of ‘architectural similarity despite topological variability,’ depicted in Fig. 1.10 [118].

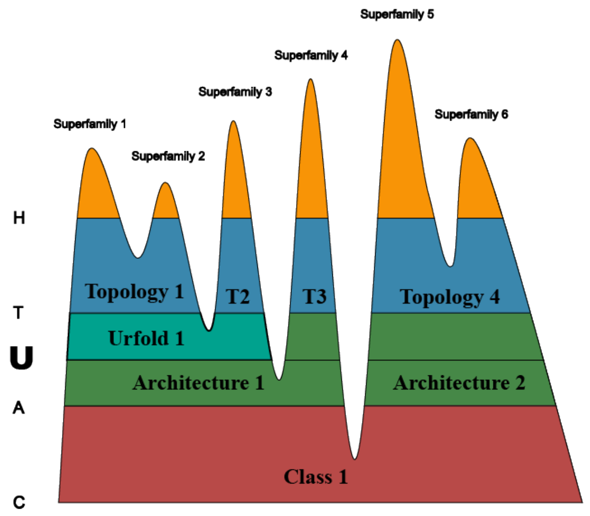


Figure 1.10: Hierarchical clustering of fold space example with CATH cut levels. Each point in fold space is a protein and all proteins in the same homologous superfamily (H; yellow) are ‘nearby.’ If the fold space is compressed to 1 dimension, densities could be seen as similar to this depiction. Finding the correct cut levels is a major challenge of hierarchical clustering. Normal CATH clustering can be seen with cut levels for Topology (blue), Architecture (green), and Class (red). However, if superfamilies 1 and 2 have similar architectures, but different topologies, they could be clustered together, in a new Urfold level, represented by a cut in between Architecture and Topology.

An example of the Urfold phenomenon can be seen in the small beta barrels: Src homology 3 (SH3) and Oligonucleotide Binding (OB) Homologous Superfamilies. Both SH3 and OB superfamilies have a similar fold and share many of the same functions, yet these similarities are obscured (or at least unacknowledged) by their being classified differently in the CATH, SCOP, and ECOD hierarchies. They both act as scaffolds in large complexes and bind other proteins on the same regions of their respective 3D structures (edge strands). While they have the same architecture, the loops between the beta strands have been permuted, resulting in different topologies. The only difference between the superfamilies at the sequence level is a shifted α helix from the N-termini to before the 4th beta strand. Because this fold can tolerate large scale mutations, such as the α helix shifting, it is extremely robust because it can fold back into the same shape [119]. The architecture has been found to be held together by a conserved hydrophobic core, which is present in both the SH3 and OB folds (see Fig. 1.11).

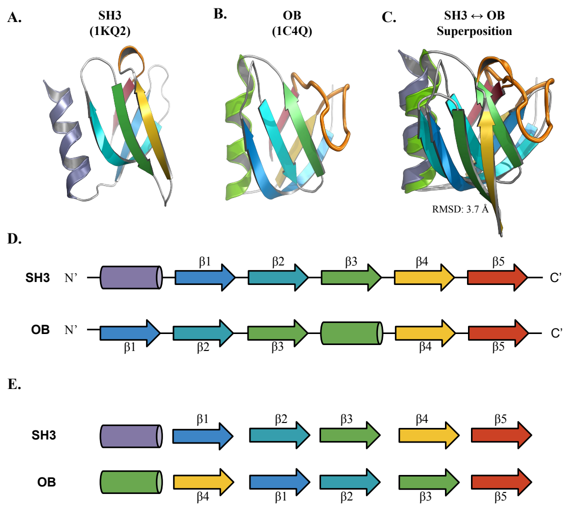


Figure 1.11: SH3 and OB Comparison in 2° and 3° structure. SH3 (A) and OB (B) structures colored by 2° structure, starting from violet to red; large loops (orange) differ between them. The superposition between SH3 and OB (C) was creating in CE/PyMOL. D) The order of 2° elements from N- to C-terminus; The only difference is shifted helix; E) The aligned 2° structures from (C).

A recent study proposed that a four stranded ancestor could have evolved into both the OB and SH3 folds by duplication followed by strand removal on the N- or C-termini respectively [120]. If this ancestor existed and was stable, it also exists within fold space. Is fold space time-resolved, where protein domains are connected via branch points in a phylogenetic network? Or is it one universal collection of all proteins, irrespective of evolutionary time. In these schemes, should the OB and SH3 folds (or, rather, 'superfolds') be connected directly (to give, e.g., an SBB as this particular urfold), or only connected at the level of their common four-stranded ancestor to give a 'proto-SBB' urfold? Most studies have only been using extant proteins, which are available from the Protein Data Bank (PDB). Ancestral state reconstructions [121, 122] will play a pivotal role in filling in the gaps of fold space, however that is extremely time consuming.

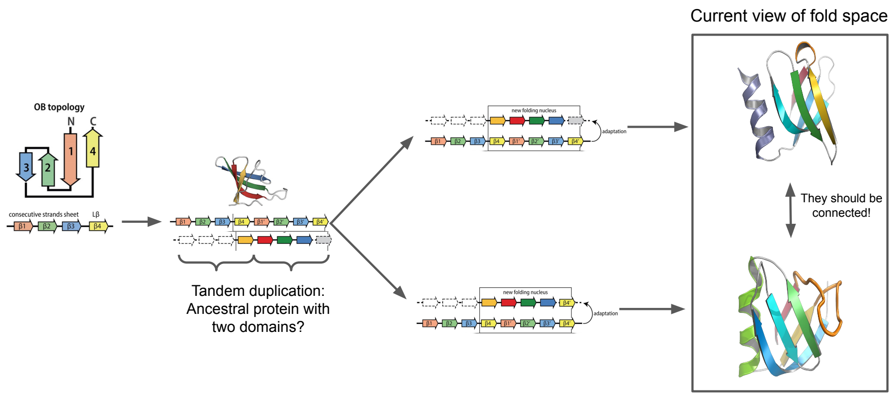


Figure 1.12: Possible Evolutionary Relationship of the OB and SH3 fold. According to [120], an ancestral OB peptide could have: (i) duplicated and (ii) removed its terminal SSE via two different, but plausible fold nuclei, (iii) creating both the SH3 and OB folds. Image modified from [120], which is under the CC-by-4.0 licence.

Through visual analysis and manual searches of the PDB, Mura et al. [118] have discovered more examples of the Urfold phenomenon, particularly in P-Loop NTPases and KH domains. Each of these examples have a common architecture and conserved hydrophobic cores, but different topologies. Do they also share common ancestors that can be duplicated and rearranged to give the permutations found in extant proteins?

In my dissertation, I will start with the hypothesis that protein fold space is continuous. I have introduced two ideas that can allow for connections between distantly related proteins in fold space: (1) common short peptide fragments (contiguous); and (2) domains with similar architectures, yet different topologies and a common hydrophobic core (non-contiguous). The reconciliation of both ideas will provide a robust common 'unit of evolution' and provide methods to bridge more remaining gaps in fold space. We hypothesize that this new 'unit of evolution' will be a structural/geometric motif (either contiguous or non-contiguous in sequence), composed of a set of residues with similar biophysical properties in some 3D/spatially conserved pattern of locations. This, in turn, will provide a new approach to identify archetypal precursors for all domain structures, and a closer understanding of the origin of life.

While we can't actually obtain full evolutionary histories for every protein, finding these new evolutionary units will enable us to target distant protein families and shine light on previously unseen connections in fold space. Any new representations of the protein universe will allow us to reimagine it, identifying patterns that we have not seen before.

1.7 Revisit the question of fold space in light of new deep learning methods

In light of new deep learning methods and latent space embeddings, the continuous vs discrete dichotomy of protein structure space must be revisited. Understanding inter-protein relationships can now be done in lower dimensional spaces, called latent space or embeddings, which can now be compared; we claim that such comparisons are more meaningful, efficacious, and–perhaps counterintuitively–more direct than are comparisons using the original input structures. Traditional all-vs-all alignments in sequence or structure space operate on the original input, making it difficult to find remote homologies. In sequence space, it is difficult to identify relationships between proteins in the 'twilight zone' with less than 25% sequence identity. New deep learning approaches overcome these challenges and can find even more distant homologies allowing us to understand protein evolution in new ways.

While the inputs will be different—sequences can be represented by strings of amino acids, and structure can be represented as contact maps, graphs, or 3D images—the output representation of the latent space remains the same. Each protein is converted to it's latent representation, a single data point, by running it through a trained model; this is commonly referred to as the 'inference' stage in ML workflows. If the dimensionality of the latent space is larger than two or three, it can be reduced for visualization purposes to 2D or 3D using either a dimensionality reduction method such as principal component analysis (PCA), an embedding approach such as t-distributed stochastic neighbor embedding (t-SNE), or the more modern uniform manifold approximation and projection (UMAP) technique. Some studies have begun to analyze the latent space, but it still remains unexplored. Typically it is colored by kingdom [83, 79], secondary structure class (all-α, all-β, and α/β) [83], discrete fold classifications (CATH, SCOP, or ECOD) [77, 87], discrete sequence families [123, 54], or discrete orthologous groups [85]. There are brief experiments of manipulating proteins in their latent/lower-dimensional representations by adding the difference of latent spaces to convert one protein into another [124, 77, 85]. One study clustered the latent space to try to reconstruct phylogenies with reasonable success [125]. However, examining latent spaces is usually more of an afterthought when analyzing models – not the end product. Downstream analysis of these new embeddings will be affected if protein structure space is considered as being discrete. A typical next task is transfer learning, where one uses the previously trained protein model (unsupervised) to predict properties at the whole protein level or residue level (supervised). Such a task is often predicting which discrete fold a protein belongs to [77, 83]. As expected, the models do not predict each discrete hierarchy well (accuracy <80%) [77, 87].

In another sense, however, the protein sequence-based community is also scrapping the discrete view in favor of a continuous view. ML algorithms used to be trained on single protein families and multiple sequence alignments obtained through iterative searches. New NLP algorithms such as Language Transformers are now using the entire Uniprot Database – where protein family sequence alignments are not needed, and there is an implicit continuity in the distribution of proteins/features. The difficult remaining issue becomes how to map results back to a discrete framework during transfer learning.

1.8 Thesis Outline

The main goal of my thesis is to explore a new representation of the protein universe to give clues about protein interrelationships, protein evolution, and the origins of life in a protein world. The proposed “Urfold” model is an innovative approach to visualize and understand patterns of similarities in the protein universe. More distant evolutionary and functional relationships can be identified when the traditional hierarchical classification scheme is thought of in this new way. Specific innovations of this thesis include algorithm and database development, such as:

- A community resource that I develop in order to enable the creation and sharing of biophysical properties and protein structures. These biophysical properties can be used as features in any machine learning model, and its use ensures that analyses are performed with standardized data rather than creating it every time.

- A novel sequence-independent, alignment-free, rotation-invariant similarity metric of proteins based on Deep Generative Models and 3D structures. This framework leverages similarities in latent-spaces rather than the 3D structures directly and encodes biophysical properties, thereby allowing higher orders of similarity to be detected (e.g. functional similarities, such as from ligand-binding pockets).

- A new approach to detect clusters, or communities, of similar protein structures using Stochastic Block Models. This method takes a different approach to clustering, allowing for proteins to span multiple clusters, thereby allowing for the continuous nature of fold space to be accounted for (rather than precluded).

The remaining 4 chapters will follow my thesis aims:

- **Chapter 2 – Aim 1: Create a database of biophysical atomic properties in 3D for the known protein universe**

    1. Develop a reproducible computational workflow to calculate biophysical and evolutionary properties for all protein domains with known experimental structures.
    2. Implement a highly distributed data service to load all proteins and biophysical properties quickly in an easily accessible API for use in any machine learning model

- **Chapter 3 – Aim 2: Build and interrogate Deep Generative Models to learn superfamily-specific geometries and properties**

    1. Train and validate Superfamily (SF) specific Variational Autoencoders (VAEs) to learn the defining geometries and biophysical properties for 20 SFs including OB, SH3, and other SFs of particular interest to our lab discovered via manual study of the literature and PDB.
    2. Explore the latent space of the 20 SF-specific VAEs to see if they capture gross structural properties such as patterns, trends, and a preference for a discrete or continuous nature of fold space.
    3. Assess the Urfold model by subjecting proteins with permuted secondary structures to the superfamily-specific VAEs.

- **Chapter 4 – Aim 3: Identify distant evolutionary relationships that bridge protein architectures and topologies that define an Urfold**

1. Identify communities of domain structures and SFs with Stochastic Block Models (SBM) when subjecting domain representatives to all 20 SF-specific models.
2. Determine the most relevant atoms while subjecting domains to SF-specific models using Layerwise-Relevance Propagation (LRP).
3. Create a tangible definition of an 'Urfold' by investigating the atomic relevance scores for domains in each SBM community to find common themes and specific examples.

- **A final conclusion on the outlook of this project and where we could go next**

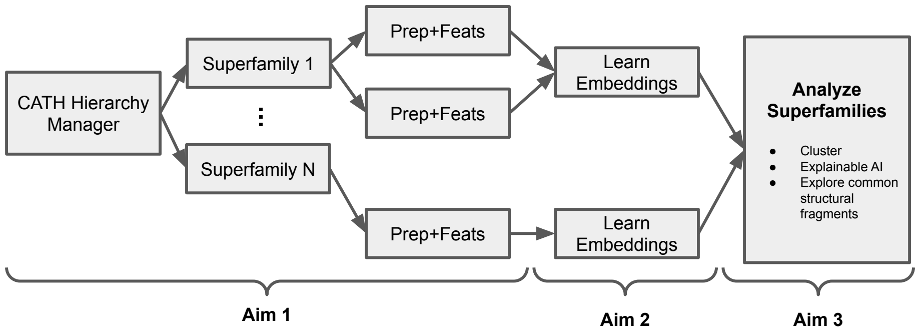


Figure 1.13: Thesis and DeepUrfold Overview. How all chapters and aims connect to each other. A dataset is first created in Aim by running multiple jobs in parallel for each CATH Homologous Superfamily and another job for each domain to Prepare Structure and Create Features (Prep+Feats). Next, the datasets created for each superfamily are used to learn embeddings for each superfamily. Finally, in Aim 3, we will analyze the embeddings from each superfamily to identify common discontiguous structural fragments with similar geometry and biophysical properties.

Bibliography

[1] Vikram Alva, Johannes Soding, and Andrei N Lupas. A vocabulary of ancient peptides at the origin of folded proteins. *eLife*, 4:e09410, Dec 2015.

[2] Eugene V Koonin, Yuri I Wolf, and Georgy P Karev. The structure of the protein universe and genome evolution. *Nature*, 420(6912):218–223, Nov 2002.

[3] J S Richardson. The anatomy and taxonomy of protein structure. *Advances in protein chemistry*, 34:167–339, 1981.

[4] Sebastian E Ahnert, Joseph A Marsh, Helena Hernández, Carol V Robinson, and Sarah A Teichmann. Principles of assembly reveal a periodic table of protein complexes. *Science*, 350(6266):aaa2245, Dec 2015.

[5] K A Dill and H S Chan. From levinthal to pathways to funnels. *Nature Structural Biology*, 4(1):10–19, Jan 1997.

[6] Guilhem Faure, Aleksey Y Ogurtsov, Svetlana A Shabalina, and Eugene V Koonin. Role of mrna structure in the control of protein folding. *Nucleic Acids Research*, 44(22):10898–10911, Dec 2016.

[7] Narayanan Eswar, Ben Webb, Marc A Marti-Renom, M S Madhusudhan, David Eramian, Min-Yi Shen, Ursula Pieper, and Andrej Sali. Comparative protein structure modeling using modeller. *Current Protocols in Bioinformatics*, Chapter 5:Unit 5.6, Oct 2006.

[8] Jianyi Yang, Renxiang Yan, Ambrish Roy, Dong Xu, Jonathan Poisson, and Yang Zhang. The i-tasser suite: protein structure and function prediction. *Nature Methods*, 12(1):7–8, Jan 2015.

[9] David E Shaw, Paul Maragakis, Kresten Lindorff-Larsen, Stefano Piana, Ron O Dror, Michael P Eastwood, Joseph A Bank, John M Jumper, John K Salmon, Yibing Shan, and Willy Wriggers. Atomic-level characterization of the structural dynamics of proteins. *Science*, 330(6002):341–346, Oct 2010.

[10] Debora S Marks, Lucy J Colwell, Robert Sheridan, Thomas A Hopf, Andrea Pagnani, Riccardo Zecchina, and Chris Sander. Protein 3d structure computed from evolutionary sequence variation. *Plos One*, 6(12):e28766, Dec 2011.

[11] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andrew J Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, Aug 2021.

[12] Anna-Leigh Brown, Minghui Li, Alexander Goncearenco, and Anna R Panchenko. Finding driver mutations in cancer: Elucidating the role of background mutational processes. *PLoS Computational Biology*, 15(4):e1006981, Apr 2019.

[13] David A Liberles, Sarah A Teichmann, Ivet Bahar, Ugo Bastolla, Jesse Bloom, Erich Bornberg-Bauer, Lucy J Colwell, A P Jason de Koning, Nikolay V Dokholyan, Julian Echave, Arne Elofsson, Dietlind L Gerloff, Richard A Goldstein, Johan A Grahnen, Mark T Holder, Clemens Lakner, Nicholas Lartillot,

Simon C Lovell, Gavin Naylor, Tina Perica, and Simon Whelan. The interface of protein structure, protein biophysics, and molecular evolution. *Protein Science*, 21(6):769–785, Jun 2012.

[14] G Apic, J Gough, and S A Teichmann. Domain combinations in archaeal, eubacterial and eukaryotic proteomes. *Journal of Molecular Biology*, 310(2):311–325, Jul 2001.

[15] Jung-Hoon Han, Sarah Batey, Adrian A Nickson, Sarah A Teichmann, and Jane Clarke. The folding and evolution of multidomain proteins. *Nature Reviews Molecular Cell Biology*, 8(4):319–330, Mar 2007.

[16] Lesley H Greene, Tony E Lewis, Sarah Addou, Alison Cuff, Tim Dallman, Mark Dibley, Oliver Redfern, Frances Pearl, Rekha Nambudiry, Adam Reid, Ian Sillitoe, Corin Yeats, Janet M Thornton, and Christine A Orengo. The cath domain structure database: new protocols and classification levels give a more comprehensive resource for exploring evolution. *Nucleic Acids Research*, 35(Database issue):D291–7, Jan 2007.

[17] Antonina Andreeva, Dave Howorth, Cyrus Chothia, Eugene Kulesha, and Alexey G Murzin. Scop2 prototype: a new approach to protein structure mining. *Nucleic Acids Research*, 42(Database issue):D310–4, Jan 2014.

[18] Naomi K Fox, Steven E Brenner, and John-Marc Chandonia. Scope: Structural classification of proteins–extended, integrating scop and astral data and classification of new structures. *Nucleic Acids Research*, 42(Database issue):D304–9, Jan 2014.

[19] Hua Cheng, R Dustin Schaeffer, Yuxing Liao, Lisa N Kinch, Jimin Pei, Shuoyong Shi, Bong-Hyun Kim, and Nick V Grishin. Ecod: an evolutionary classifi-

cation of protein domains. *PLoS Computational Biology*, 10(12):e1003926, Dec 2014.

[20] Robert D Finn, Alex Bateman, Jody Clements, Penelope Coggill, Ruth Y Eberhardt, Sean R Eddy, Andreas Heger, Kirstie Hetherington, Liisa Holm, Jaina Mistry, Erik L L Sonnhammer, John Tate, and Marco Punta. Pfam: the protein families database. *Nucleic Acids Research*, 42(Database issue):D222–30, Jan 2014.

[21] Timothy A Holland, Stella Veretnik, Ilya N Shindyalov, and Philip E Bourne. Partitioning protein structures into domains: why is it so difficult? *Journal of Molecular Biology*, 361(3):562–590, Aug 2006.

[22] Ian Sillitoe, Natalie Dawson, Tony E Lewis, Sayoni Das, Jonathan G Lees, Paul Ashford, Adeyelu Tolulope, Harry M Scholes, Ilya Senatorov, Andra Bujan, Fatima Ceballos Rodriguez-Conde, Benjamin Dowling, Janet Thornton, and Christine A Orengo. Cath: expanding the horizons of structure-based functional annotations for genome sequences. *Nucleic Acids Research*, 47(D1):D280–D284, Jan 2019.

[23] R Dustin Schaeffer, Lisa N Kinch, Jimin Pei, Kirill E Medvedev, and Nick V Grishin. Completeness and consistency in structural domain classifications. *ACS omega*, 6(24):15698–15707, Jun 2021.

[24] Jorge Luis Borges. *The Library of Babel*. David R Godine Pub, 2000.

[25] C B Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230, Jul 1973.

[26] Ruslan I Sadreyev, Bong-Hyun Kim, and Nick V Grishin. Discrete-continuous duality of protein structure space. *Current Opinion in Structural Biology*, 19(3):321–328, Jun 2009.

[27] Jeffrey Skolnick, Adrian K Arakaki, Seung Yup Lee, and Michal Brylinski. The continuity of protein structure space is an intrinsic property of proteins. *Proceedings of the National Academy of Sciences of the United States of America*, 106(37):15690–15695, Sep 2009.

[28] L Holm and C Sander. Mapping the protein universe. *Science*, 273(5275):595–603, Aug 1996.

[29] Rachel Kolodny, Sergey Nepomnyachiy, Dan S Tawfik, and Nir Ben-Tal. Bridging themes: short protein segments found in different architectures. *Molecular Biology and Evolution*, 38(6):2191–2208, May 2021.

[30] Sergey Nepomnyachiy, Nir Ben-Tal, and Rachel Kolodny. Global view of the protein universe. *Proceedings of the National Academy of Sciences of the United States of America*, 111(32):11691–11696, Aug 2014.

[31] Jingtong Hou, Se-Ran Jun, Chao Zhang, and Sung-Hou Kim. Global mapping of the protein structure space and application in structure-based inference of protein function. *Proceedings of the National Academy of Sciences of the United States of America*, 102(10):3651–3656, Mar 2005.

[32] Margarita Osadchy and Rachel Kolodny. Maps of protein structure space reveal a fundamental relationship between protein structure and function. *Proceedings of the National Academy of Sciences of the United States of America*, 108(30):12301–12306, Jul 2011.

[33] Yana Bromberg, Ariel A. Aptekmann, Yannick Mahlich, Linda Cook, Stefan Senn, Maximillian Miller, Vikas Nanda, Diego U. Ferreiro, and Paul G. Falkowski. Quantifying structural relationships of metal-binding sites suggests origins of biological electron transfer. *Science Advances*, 8(2), Jan 2022.

[34] Sergey Nepomnyachiy, Nir Ben-Tal, and Rachel Kolodny. Complex evolutionary footprints revealed in an analysis of reused protein segments of diverse lengths. *Proceedings of the National Academy of Sciences of the United States of America*, 114(44):11703–11708, Oct 2017.

[35] B L Sibanda and J M Thornton. Beta-hairpin families in globular proteins. *Nature*, 316(6024):170–174, Jul 1985.

[36] M J E Sternberg and J M Thornton. On the conformation of proteins: The handedness of the β-strand-α-helix-β-strand unit. *Journal of Molecular Biology*, 105(3):367–382, Aug 1976.

[37] R Unger, D Harel, S Wherland, and J L Sussman. A 3d building blocks approach to analyzing and predicting structure of proteins. *Proteins*, 5(4):355–373, 1989.

[38] Philipp Bucher, Kevin Karplus, Nicolas Moeri, and Kay Hofmann. A flexible motif search technique based on generalized profiles. *Computers & Chemistry*, 20(1):3–23, Mar 1996.

[39] K F Han and D Baker. Global properties of the mapping between local amino acid sequence and local structure in proteins. *Proceedings of the National Academy of Sciences of the United States of America*, 93(12):5814–5818, Jun 1996.

[40] G J Kleywegt. Recognition of spatial motifs in protein structures. *Journal of Molecular Biology*, 285(4):1887–1897, Jan 1999.

[41] I N Shindyalov and P E Bourne. An alternative view of protein fold space. *Proteins*, 38(3):247–260, Feb 2000.

[42] Alexandre De Brevern, Anne-Claude Camproux, Serge Hazout, Catherine Etchebest, and Pierre Tufféry. Protein structural alphabets: beyond the secondary structure description. *Recent research developments in protein engineering*, Jan 2001.

[43] Agnel Praveen Joseph, Garima Agarwal, Swapnil Mahajan, Jean-Christophe Gelly, Lakshmipuram S Swapna, Bernard Offmann, Frédéric Cadet, Aurélie Bornot, Manoj Tyagi, Hélène Valadié, Bohdan Schneider, Catherine Etchebest, Narayanaswamy Srinivasan, and Alexandre G De Brevern. A short survey on protein blocks. *Biophysical reviews*, 2(3):137–147, Aug 2010.

[44] Inbal Budowski-Tal, Yuval Nov, and Rachel Kolodny. Fragbag, an accurate representation of protein structure, retrieves structural neighbors from the entire pdb quickly and accurately. *Proceedings of the National Academy of Sciences of the United States of America*, 107(8):3481–3486, Feb 2010.

[45] Rachel Kolodny, Patrice Koehl, Leonidas Guibas, and Michael Levitt. Small libraries of protein fragments model native protein structures accurately. *Journal of Molecular Biology*, 323(2):297–307, Oct 2002.

[46] Iddo Friedberg and Adam Godzik. Connecting the protein structure universe by using sparse recurring fragments. *Structure*, 13(8):1213–1224, Aug 2005.

[47] Elena Zotenko, Rezarta Islamaj Dogan, W John Wilbur, Dianne P O'Leary, and Teresa M Przytycka. Structural footprinting in protein structure comparison: the impact of structural fragments. *BMC Structural Biology*, 7:53, Aug 2007.

[48] Antonina Andreeva, Andreas Prlić, Tim J P Hubbard, and Alexey G Murzin. Sisyphus–structural alignments for proteins with non-trivial relationships. *Nucleic Acids Research*, 35(Database issue):D253–9, Jan 2007.

[49] Narcis Fernandez-Fuentes, Joseph M Dybas, and Andras Fiser. Structural characteristics of novel protein folds. *PLoS Computational Biology*, 6(4):e1000750, Apr 2010.

[50] Alexander Goncearenco and Igor N Berezovsky. Prototypes of elementary functional loops unravel evolutionary connections between protein functions. *Bioinformatics*, 26(18):i497–503, Sep 2010.

[51] Yang Liu, Qing Ye, Liwei Wang, and Jian Peng. Learning structural motif representations for efficient protein structure search. *Bioinformatics*, 34(17):i773–i780, Sep 2018.

[52] Philippe Youkharibache. Protodomains: Symmetry-related supersecondary structures in proteins and self-complementarity. *Methods in Molecular Biology*, 1958:187–219, 2019.

[53] Sebastian Bittrich, Stephen K Burley, and Alexander S Rose. Real-time structural motif searching in proteins using an inverted index strategy. *PLoS Computational Biology*, 16(12):e1008502, Dec 2020.

[54] Janani Durairaj, Mehmet Akdel, Dick de Ridder, and Aalt D J van Dijk. Geometricus represents protein structures as shape-mers derived from moment invariants. *Bioinformatics*, $36(Suppl_2) : i718–i725, Dec2020$.

[55] Andrew Harrison, Frances Pearl, Richard Mott, Janet Thornton, and Christine Orengo. Quantifying the similarities within fold space. *Journal of Molecular Biology*, 323(5):909–926, Nov 2002.

[56] S F Altschul, T L Madden, A A Schäffer, J Zhang, Z Zhang, W Miller, and D J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, Sep 1997.

[57] Russell Doolittle. *Of Urfs and Orfs: A Primer on How to Analyze Derived Amino Acid Sequences*. University Science Books, 1986.

[58] B Rost. Twilight zone of protein sequence alignments. *Protein Engineering Design and Selection*, 12(2):85–94, Feb 1999.

[59] S B Needleman and C D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, Mar 1970.

[60] T F Smith and M S Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, Mar 1981.

[61] S Henikoff and J G Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 89(22):10915–10919, Nov 1992.

[62] D J Lipman and W R Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, Mar 1985.

[63] S F Altschul, W Gish, W Miller, E W Myers, and D J Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, Oct 1990.

[64] Martin Steinegger and Johannes Soding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, Nov 2017.

[65] K Karplus, C Barrett, and R Hughey. Hidden markov models for detecting remote protein homologies. *Bioinformatics*, 14(10):846–856, 1998.

[66] A Krogh, M Brown, I S Mian, K Sjolander, and D Haussler. Hidden markov models in computational biology. applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, Feb 1994.

[67] R Hughey and A Krogh. Hidden markov models for sequence analysis: extension and analysis of the basic method. *Computer applications in the biosciences: CABIOS*, 12(2):95–107, Apr 1996.

[68] Sean R Eddy. Accelerated profile hmm searches. *PLoS Computational Biology*, 7(10):e1002195, Oct 2011.

[69] Martin Steinegger, Markus Meier, Milot Mirdita, Harald Vohringer, Stephan J Haunsberger, and Johannes Soding. Hh-suite3 for fast remote homology detection and deep protein annotation. *BMC Bioinformatics*, 20(1):473, Sep 2019.

[70] L Steven Johnson, Sean R Eddy, and Elon Portugaly. Hidden markov model speed heuristic and iterative hmm search procedure. *BMC Bioinformatics*, 11:431, Aug 2010.

[71] Faruck Morcos, Andrea Pagnani, Bryan Lunt, Arianna Bertolino, Debora S Marks, Chris Sander, Riccardo Zecchina, José N Onuchic, Terence Hwa, and Martin Weigt.

Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proceedings of the National Academy of Sciences of the United States of America*, 108(49):E1293–301, Dec 2011.

[72] Thomas A Hopf, John B Ingraham, Frank J Poelwijk, Charlotta P I Schärfe, Michael Springer, Chris Sander, and Debora S Marks. Mutation effects predicted from sequence co-variation. *Nature Biotechnology*, 35(2):128–135, Feb 2017.

[73] Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, Wei Xie, Gail L Rosen, Benjamin J Lengerich, Johnny Israeli, Jack Lanchantin, Stephen Woloszynek, Anne E Carpenter, Avanti Shrikumar, Jinbo Xu, Evan M Cofer, and Casey S Greene. Opportunities and obstacles for deep learning in biology and medicine. *Journal of the Royal Society, Interface*, 15(141), Apr 2018.

[74] UniProt Consortium. Uniprot: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, Jan 2021.

[75] Martin Steinegger, Milot Mirdita, and Johannes Soding. Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nature Methods*, 16(7):603–606, Jul 2019.

[76] Martin Steinegger and Johannes Soding. Clustering huge protein sequence sets in linear time. *Nature Communications*, 9(1):2542, Jun 2018.

[77] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 12(6):654–669.e3, Jun 2021.

[78] Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Pro-

tein design and variant prediction using autoregressive generative models. *Nature Communications*, 12(1):2403, Apr 2021.

[79] Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature Methods*, 15(10):816–822, Oct 2018.

[80] Jeanne Trinquier, Guido Uguzzoni, Andrea Pagnani, Francesco Zamponi, and Martin Weigt. Efficient generative modeling of protein sequences using simple autoregressive models. *Nature Communications*, 12(1):5800, Oct 2021.

[81] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, 16(12):1315–1322, Dec 2019.

[82] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S. Song. Evaluating protein transfer learning with tape. *BioRxiv*, Jun 2019.

[83] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, DEBSINDHU Bhowmik, and Burkhard Rost. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *BioRxiv*, Jul 2020.

[84] Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R. Eguchi, Po-Ssu Huang, and Richard Socher. Progen: language modeling for protein generation. *BioRxiv*, Mar 2020.

[85] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological

structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences of the United States of America*, 118(15), Apr 2021.

[86] Tymor Hamamsy, James T. Morton, Daniel Berenberg, Nicholas Carriero, Vladimir Gligorijevic, Robert Blackwell, Charlie E. M. Strauss, Julia Koehler Leman, Kyunghyun Cho, and Richard Bonneau. Tm-vec: template modeling vectors for fast homology detection and alignment. *BioRxiv*, Jul 2022.

[87] Michael Heinzinger, Maria Littmann, Ian Sillitoe, Nicola Bordin, Christine Orengo, and Burkhard Rost. Contrastive learning on protein embeddings enlightens midnight zone at lightning speed. *BioRxiv*, Nov 2021.

[88] Jamie Morton, Charlie Strauss, Robert Blackwell, Daniel Berenberg, Vladimir Gligorijevic, and Richard Bonneau. Protein structural alignments from sequence. *BioRxiv*, Nov 2020.

[89] Vamsi Nallapareddy, Nicola Bordin, Ian Sillitoe, Michael Heinzinger, Maria Littmann, Vaishali Waman, Neeladri Sen, Burkhard Rost, and Christine Orengo. Cathe: Detection of remote homologues for cath superfamilies using embeddings from protein language models. *BioRxiv*, Mar 2022.

[90] W Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, Sep 1976.

[91] W Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 34(5):827–828, Sep 1978.

[92] Christine A. Orengo and William R. Taylor. *[36] SSAP: Sequential structure alignment program for protein structure comparison*, volume 266 of *Methods in Enzymology*, page 617–635. Elsevier, 1996.

[93] L Holm and C Sander. Dali: a network tool for protein structure comparison. *Trends in Biochemical Sciences*, 20(11):478–480, Nov 1995.

[94] I N Shindyalov and P E Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Engineering Design and Selection*, 11(9):739–747, Sep 1998.

[95] J F Gibrat, T Madej, and S H Bryant. Surprising similarities in structure comparison. *Current Opinion in Structural Biology*, 6(3):377–385, Jun 1996.

[96] Adam Zemla. Lga: A method for finding 3d similarities in protein structures. *Nucleic Acids Research*, 31(13):3370–3374, Jul 2003.

[97] Yang Zhang and Jeffrey Skolnick. Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic Acids Research*, 33(7):2302–2309, Apr 2005.

[98] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins*, 57(4):702–710, Dec 2004.

[99] Mathilde Carpentier and Jacques Chomilier. Protein multiple alignments: sequence-based versus structure-based programs. *Bioinformatics*, 35(20):3970–3980, Oct 2019.

[100] Runze Dong, Zhenling Peng, Yang Zhang, and Jianyi Yang. mtm-align: an algorithm for fast and accurate multiple protein structure alignment. *Bioinformatics*, 34(10):1719–1725, May 2018.

[101] Michel van Kempen, Stephanie Kim, Charlotte Tumescheit, Milot Mirdita, Johannes Soeding, and Martin Steinegger. Foldseek: fast and accurate protein structure search. *BioRxiv*, Feb 2022.

[102] Matthew Menke, Bonnie Berger, and Lenore Cowen. Matt: local flexibility aids protein multiple structure alignment. *PLoS Computational Biology*, 4(1):e10, Jan 2008.

[103] Maksim V Shegay, Dmitry A Suplatov, Nina N Popova, Vytas K Švedas, and Vladimir V Voevodin. parmatt: parallel multiple alignment of protein 3d-structures with translations and twists for distributed-memory systems. *Bioinformatics*, 35(21):4456–4458, Nov 2019.

[104] Mehmet Akdel, Janani Durairaj, Dick de Ridder, and Aalt D J van Dijk. Caretta - a multiple protein structure alignment and feature extraction suite. *Computational and structural biotechnology journal*, 18:981–992, Apr 2020.

[105] Cameron Mura, Eli J Draizen, and Philip E Bourne. Structural biology meets data science: does anything change? *Current Opinion in Structural Biology*, 52:95–102, Oct 2018.

[106] Raphael J. L. Townshend, Martin Vogele, Patricia Suriana, Alexander Derry, Alexander Powers, Yianni Laloudakis, Sidhika Balachandar, Bowen Jing, Brandon Anderson, Stephan Eismann, Risi Kondor, Russ B. Altman, and Ron O. Dror. Atom3d: Tasks on molecules in three dimensions. *arXiv*, Dec 2020.

[107] Afshine Amidi, Shervine Amidi, Dimitrios Vlachakis, Vasileios Megalooikonomou, Nikos Paragios, and Evangelia I Zacharaki. Enzynet: enzyme classification using 3d convolutional neural networks on spatial representation. *PeerJ*, 6:e4750, May 2018.

[108] Joshua Hochuli, Alec Helbling, Tamar Skaist, Matthew Ragoza, and David Ryan Koes. Visualizing convolutional neural network protein-ligand scoring. *Journal of molecular graphics & modelling*, 84:96–108, Sep 2018.

[109] J Jiménez, S Doerr, G Martínez-Rosell, A S Rose, and G De Fabritiis. Deepsite: protein-binding site predictor using 3d-convolutional neural networks. *Bioinformatics*, 33(19):3036–3042, Oct 2017.

[110] Ali Tugrul Balci, Can Gumeli, Asma Hakouz, Deniz Yuret, Ozlem Keskin, and Attila Gursoy. Deepinterface: Protein-protein interface validation using 3d convolutional neural networks. *BioRxiv*, Apr 2019.

[111] Nicolas Renaud, Cunliang Geng, Sonja Georgievska, Francesco Ambrosetti, Lars Ridder, Dario F Marzella, Manon F Réau, Alexandre M J J Bonvin, and Li C Xue. Deeprank: a deep learning framework for data mining 3d protein-protein interfaces. *Nature Communications*, 12(1):7068, Dec 2021.

[112] Raphael J. L. Townshend, Rishi Bedi, and Ron O. Dror. Transferrable end-to-end learning for protein interface prediction. *arXiv*, Jul 2018.

[113] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. *Advances in Neural Information Processing Systems*, 2017.

[114] John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. *Advances in Neural Information Processing Systems*, 2019.

[115] Manon Réau, Nicolas Renaud, Li C. Xue, and Alexandre M.J.J Bonvin. Deeprank-gnn: A graph neural network framework to learn patterns in protein-protein interfaces. *BioRxiv*, Dec 2021.

[116] Bowen Jing, Stephan Eismann, Pratham N. Soni, and Ron O. Dror. Equivariant graph neural networks for 3d macromolecular structure. *arXiv*, Jun 2021.

[117] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. *arXiv*, Feb 2021.

[118] Cameron Mura, Stella Veretnik, and Philip E Bourne. The urfold: Structural similarity just above the superfold level? *Protein Science*, 28(12):2119–2126, Nov 2019.

[119] Philippe Youkharibache, Stella Veretnik, Qingliang Li, Kimberly A Stanek, Cameron Mura, and Philip E Bourne. The small β-barrel domain: A survey-based structural analysis. *Structure*, 27(1):6–26, Jan 2019.

[120] Claudia Alvarez-Carreño, Petar I Penev, Anton S Petrov, and Loren Dean Williams. Fold evolution before luca: Common ancestry of sh3 domains and ob domains. *Molecular Biology and Evolution*, 38(11):5134–5143, Oct 2021.

[121] Georg K A Hochberg and Joseph W Thornton. Reconstructing ancient proteins to understand the causes of structure and function. *Annual review of biophysics*, 46:247–269, May 2017.

[122] Avery G A Selberg, Eric A Gaucher, and David A Liberles. Ancestral sequence reconstruction: from chemical paleogenetics to maximum likelihood algorithms and beyond. *Journal of Molecular Evolution*, 89(3):157–164, Apr 2021.

[123] Mehmet Akdel, Douglas EV Pires, Eduard Porta-Pardo, Jurgen Janes, Arthur O Zalevsky, Balint Meszaros, Patrick Bryant, Lydia L Good, Roman A Laskowski,

Gabriele Pozzati, Aditi Shenoy, Wensi Zhu, Petras Kundrotas, Victoria Ruiz-Serra, Carlos HM Rodrigues, Alistair S Dunham, David Burke, Neera Borkakoti, Sameer Velankar, Adam Frost, and Pedro Beltrao. A structural biology community assessment of alphafold 2 applications. *BioRxiv*, Sep 2021.

[124] Tristan Bepler and Bonnie Berger. Learning protein sequence embeddings using information from structure. *arXiv*, Feb 2019.

[125] Xinqiang Ding, Zhengting Zou, and Charles L Brooks III. Deciphering protein evolution and fitness landscapes with latent space models. *Nature Communications*, 10(1):5644, Dec 2019.

Chapter 2

Prop3D: A flexible, Python-based platform for protein structural properties and biophysical data in machine learning

Eli J. Draizen[1,2*], Luis Felipe R. Murillo[3], John Readey[4], Cameron Mura[2*], Philip E. Bourne[1,2*]

1 Department of Biomedical Engineering; University of Virginia; Charlottesville, VA; USA

2 School of Data Science; University of Virginia; Charlottesville, VA; USA

3 Department of Anthropology; University of Notre Dame; South Bend, IN; USA

4 The HDF Group; Champaign, IL; USA

*Correspondence: ed4bu@virginia.edu, cmura@virginia.edu, peb6a@virginia.edu

Abstract

Machine learning has a rich history in structural bioinformatics, and modern approaches, such as deep learning, are revolutionizing our knowledge of the subtle relationships between biomolecular sequence, structure, function, dynamics and evolution. As with any advance that rests upon statistical learning approaches, the recent progress in biomolecular sciences is enabled by the availability of vast volumes of sufficiently-variable data. To be of utility, such datasets must be well-structured, intelligible/manipulable, machine-parseable, and so on; these and related challenges become especially acute at scale. In structural bioinformatics, such data generally relate to protein three-dimensional (3D) structures. A significant and often recurring challenge concerns the creation of large, high-quality, openly-accessible datasets that can be used for specific training/benchmarking tasks in machine learning pipelines and predictive modeling projects, along with reproducible splits for training and testing. Here, we report Prop3D, a protein biophysical and evolutionary featurization and data-processing pipeline that we developed and deployed (both in the cloud and on local HPC resources) in order to systematically and reproducibly create comprehensive datasets, using the Highly Scalable Data Service (HSDS). Prop3D can be of broader utility for other structure-related workflows as a community-wide resource, particularly for tasks that arise at the intersection of deep learning and classical structural bioinformatics.

Author summary

We have developed a 'Prop3D' platform, and associated 'Prop3D-20sf' protein dataset, to allow the creation, sharing, and reuse of atomically-resolved biophysical properties for any library of protein domains, e.g. all of those found in CATH. Our workflow can be deployed on various computational platforms (cloud-based or local high/performance compute clusters), and scalability is achieved largely by saving the results to distributed HDF5 files using the Highly Scalable Data Storage (HSDS) service. The datasets and data-splits that we provide (using HSDS) can be freely accessed via a standard representational state transfer (ReST) application programming interface (API), along with accompanying Python wrappers for NumPy and the popular ML framework PyTorch.

2.1 Introduction

The advent of AlphaFold2 [1] and related deep learning approaches now enables us to access the 3D structure of virtually any protein sequence. As was the case for sequence-level data in the 1980s-2000s (enabled by technologies such as PCR and, ultimately, genome sequencing), 3D structural data has now been transformed into a readily available commodity. How might such a wealth of structural data inform our understanding of biology's central *sequence* $\leftrightarrow$ *structure* $\leftrightarrow$ *function* paradigm? Two new, post-AlphaFold2 challenges can be identified: (i) elucidating the *relationships* between all structures in the protein universe, and (ii) armed with millions of new protein structures [2], exploring the limits of protein *function prediction*. Arguably, structural bioinformatics approaches should now be an even more powerful tool in analyzing and accurately predicting protein function.

In structural bioinformatics, the ‘data’ center around protein 3D structures. In this work, we take such ‘data’ to mean the geometric structures themselves, augmented by a multitude of other properties. These other properties can be (i) at potentially varying length-scales (atomic, residue-level, domains, etc.), and (ii) of numerous types, both *biological* (e.g., phylogenetic conservation at a site) and *physicochemical* (e.g., hydrophobicity or partial charge of an atom, concavity of a patch of surface residues, etc.). A significant and persistent challenge in developing and deploying ML workflows in structural bioinformatics concerns the availability of large, high-quality, openly-accessible datasets that can be (easily) used in large-scale ML and predictive modeling projects. Here, ‘high-quality’ implies that specific training/benchmarking tasks can be performed reproducibly and without undue effort, and that the data splits for model training/testing/validation are reproducible. (A stronger requirement is that the split method also be at least semi-plausible, or not nonsensical, in terms of the underlying biology of a system—e.g., taking into account evolutionary relationships that muddle the assumed [statistical] independence of the splits; we discuss this below.)

In classical bioinformatics, transferring functional annotations from a protein with a common evolutionary history to a protein of interest is a frequent task. A conventional approach to this task typically applies sequence or structure comparison (e.g., via BLAST [3] or TM-Align [4], respectively) of a protein of interest to a database of all known proteins, followed by a somewhat manual process of ‘copying’ the previously annotated function into a new database record for the protein of interest. However, in the era of ML, we can now try to go automatically from sequence or structure directly to functional annotations; the ML models can ‘learn’ these evolutionary relationships between proteins as part of the model, removing the more manual/alignment-related steps.

However, ML workflows working with proteins—and, in particular, protein 3D structures—are far more challenging, from a technological and data-engineering perspective, than are many of the standard and more routine ML workflows that have been designed to work with inputs in other domains (a text corpus in natural language processing, image data for classification tasks, etc.). Proteins are more difficult to work with, from both a basic and applied ML perspective, for several types of reasons, including: (i) all proteins are related through evolution, thereby causing 'data leakage' [5]; (ii) raw/unprocessed protein structures are not always biophysically and chemically well-formed (e.g., atoms or entire residues may be missing) [6, 7]; (iii) somewhat related, some protein structures may 'stress-test' existing data structures by having, for instance, multiple rotamers/conformers at some sites; (iv) biophysical properties, which aren't always included/learned, are just as critical (if not more so) as the 3D geometry itself; and (v) there are many different possible representational approaches/models of protein structures (volumetric data, graphs, etc.) that can yield different results. In short, protein data must be carefully inspected/processed before they can be successfully used and split in precise ways to create robust ML models.

In this paper we present Prop3D, a new protein domain structure dataset with cleaned/prepared structures, pre-calculated biophysical and evolutionary properties, and different protein representations, along with train/test splits. We also include methods to recreate the dataset in a distributed manner and read in Prop3D for use in machine learning models. We describe this new software and its associated dataset (Prop3D-20sf), after first delineating some of the specific considerations that motivated and shaped Prop3D's design.

2.2 Motivating factors: Data leakage, biophysical properties, and protein representations

2.2.1 Evolutionary data leakage

ML with proteins is uniquely challenging because all naturally occurring proteins are interrelated via the biological processes of molecular evolution [8]. Therefore, randomly chosen train/test splits are meaningless, as there is bound to be crossover, ultimately leading to overfitting of the model. Moreover, the available datasets are biased—they sample the protein universe in a highly non-uniform (or, rather, *non-representative*) manner (Figure 2.1)—which leads to biased models. For example, there are simply more 3D structures available in the Protein Data Bank (PDB [9]) for certain protein superfamilies because, for instance, some of those families were of specific (historical) interest to specific laboratories, certain types of proteins are more intrinsically amenable to crystallization (e.g., lysozyme), some might have been disproportionately more studied and structurally characterized because they are drug targets (e.g., kinases), certain protein families were preferentially selected for during evolution [10], and so on.

A common approach to handle this type of bias is to create training and validating splits that ensure that no protein with $\geq 20\%$ sequence identify is on the same side of the split[11].

In training ML models at the level of full (intact) protein chains, another source of bias in constructing training and validation sets stems from the phenomenon of domain re-use. This is an issue because many full protein chains are multi-domain (particularly true for proteins $\gtrapprox$ 120-150 residues), and many of those individual

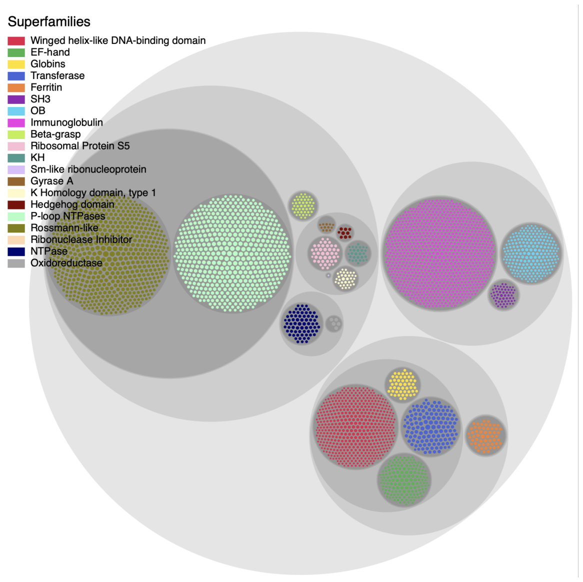


Figure 2.1: Uneven distribution of protein superfamiles. We show 20 superfamilies of interest to show how the number of known domains structures varies between superfamilies. The CATH hierarchy is shown as a circle packing chart.

domains can share similar 3D structures (and functions) and be grouped, themselves, into distinct superfamilies. Some multi-domain proteins contain multiples of a given protein domain and so on; in other words, full-length proteins generally evolved so as to utilize individual domains in a highly modular manner (Figure 2.2). While splitting based on 20% sequence identity does limit this problem to some extent (if two domains have less than that level of similarity but are still from the same superfamily), a simple, straight-ahead split at a 20% (or whatever threshold) might negatively impact an ML algorithm at the very fundamental level of model training . In principle, note that this problem could also hold at the finer scale of shared (sub-domain–level) structural fragments, too (giving an even more difficult problem).

A possible approach to mitigate this type bias is to (i) create 'one-class' superfamily-specific models; or (ii) create multi-superfamily models, making sure to (a) over-sample proteins from under-represented classes; and (b) under-sample proteins from

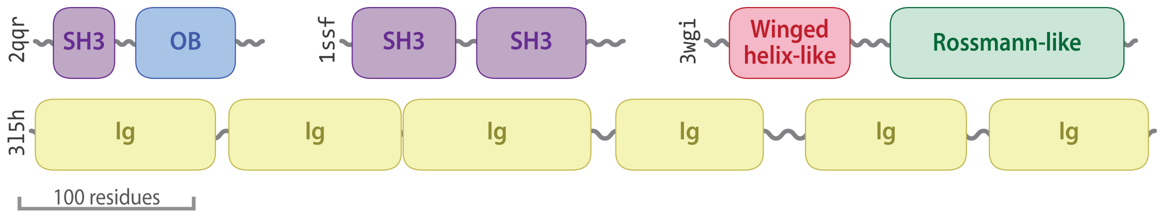


Figure 2.2: Data leakage and multi-domain proteins: A prime example of evolution-induced data leakage stems from the modular anatomy of many proteins, wherein multiple copies (which often only slightly vary—i.e., paralogous) of a particular domain are stitched together as part of the overall protein. This phenomenon is particularly prevalent among protein homologs from more phylogenetically recent species (e.g., eukaryotes like human or yeast, versus archaea or bacterial lineages). For example, many proteins that contain SH3, OB and Ig domains are found to include multiple copies of those domains.

over-represented classes [12, 5].

2.2.2 Biophysical properties

In many ML problems on proteins, is it useful to include biophysical properties mapped onto 3D locations of atoms and residues. However, sometime they are ignored as in sequence-based methods, which ignore the structure entirely, often only using a one-hot encoding of the sequence and maybe some evolutionary information. In other cases, 3D structures are used and only the geometry is used as input, neglecting the crucial biophysical properties that help define a protein's physiological function. There is also a trend in ML wherein one lets the model create its own embeddings, using only a small amount of hand-curated data (e.g., only atom type). Such approaches are generally taken because (i) it is expensive to calculate a full suite of biophysical properties for every atom, say on the scale of the entire PDB (≈200K structures); and (ii) the available models, theories and computational formalisms used to describe the biophysical properties of proteins (e.g., approximate electrostatics models, such as the generalized Born) may be insufficiently accurate, thereby adversely influencing the resultant ML models.

Irrespective of the specific details of one use-case or set of tasks versus another, it remains useful to have available a database of pre-calculated biophysical properties. Such a database would enable one to: (i) save time during the ML training process, by avoiding repetition of calculations that many others in the community may have already performed on exactly the same proteins (note that this also speaks to the key issue of reproducibility of an ML workflow or bioinformatics pipeline); and (ii) compare the predicted embeddings of the ML model to known biophysical properties, thereby providing a way to assess the accuracy and veracity of the ML model under development.

Several available databases offer biophysical properties of proteins at different 'levels' (atomic, residue, etc.) of structure, as shown in Table 2.1.

2.2.3 Protein representations

There are many different ways to represent a protein for use in ML, each with its own strengths and weaknesses. Many protein structure & feature databases are 'hard-wired' so as to include data that can populate only one type of representation; however, to be flexible and agile (and therefore more usable), new databases need to allow easy methods to switch between various alternate representations of proteins. The remainder of this section describes approaches that have been used, wherein a protein is represented as a simple sequence, as a graph-based model (residue-residue contact networks), or as a 3D volumetric dataset. We now briefly consider each of these in turn.

Dataset	Wikidata	Domain Level	Residue Level	Atom Level	Residue-Residue Graph	2° Structure	Electrostatics & Charge	Surface & Curvature	Protein Interaction Sites	Train/Validation Splits	Clusters	Evolutionary Info	File format
PDB [9]	Q766195		✓	✓†		✓		✓			✓		Web, PDB & MMTF
UniProt [13]	Q905695		✓			✓					✓		Web & ReST
CATH [14]	Q5008897	✓	✓	✓†							✓	✓	PDB & ReST
FEATURE [15]	Q114878648		✓	✓		✓	✓	✓					ASCII
PredictProtein [16]	Q7239681		✓			✓		✓	✓			✓	Web, ReST & JSON
DescribePROT [17]	Q111288739		✓					✓	✓			✓	Web & JSON
ATOM3D/DIPS [18]	Q114878673		✓	✓†					✓	✓	✓		JSON & PyTorch
ProteinNet [19]	Q114878717		✓	✓†		✓				✓	✓	✓	TensorFlow
SidechainNet [20]	Q114878822		✓	✓†		✓				✓	✓	✓	PyTorch & Pickle
Prop3D [this work]	**Q108040542**	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	**HDF, HSDS & PyTorch**

Table 2.1: Protein Feature Datasets for Machine Learning. Many different datasets of sequences, structures, and biophysical properties exist, but all contain different amounts data, different levels of data (chain, domain, residue, atom), and some contain biophysical properties attached to each atom and/or residue. The † denotes a database that uses atomic coordinates, but with no biophysical properties associated with those coordinates.

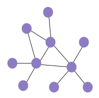
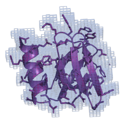
Dimensionality	**Representation**	**Example**
1D	Protein Sequence	MIANE...
2D	Residue-Residue Graph	
3D	Protein Structure 3D Volume	

Table 2.2: Protein Structure Representations

Protein Sequences

The simplest approach to represent a protein is to treat it as a sequence of amino acids, ignoring all structural information. In ML workflows, the sequence is generally "one-hot encoded", meaning that each individual character(/residue) in the sequence is attributed with a 20-element vector; in that vector, all elements are set to zero except for the index of the amino acid type that matches the current position, which is set to one. Biophysical properties can also be appended to this representation, giving a feature vector.

Residue-Residue Graph

A conceptually straightforward method to capture a 3D protein structure is to build a graph, using the amino acid residues as vertices and contacts between those residues (near in 3D space) as edges. Individual nodes can be attributed with the 'one-hot' encoding of the residue along with biophysical properties, and to each edge can be attributed geometric properties such as a simple Euclidean distance (e.g., between the two residues/nodes), an angle of interest (defined by three atoms), any dihedral angles that one likes (defined by four atoms), and so on. These graphs can be fully connected, e.g. all residues are connected to one another, or they may include edges only between residues that lie within a certain cutoff distance of one another (e.g. 5).

Protein Structures as 3D Volumetric Data

Another approach to handle protein structures in ML is to treat them as spatially discretized 3D images, wherein volumetric elements (‘voxels’) that intersect with an atom are attributed with biophysical properties of the overlapping atom. Here, note that one must define ‘*an atom*’ precisely, e.g. as a sphere of a given van der Waals radius, centered at a specific point in space (the atom’s coordinates), such that the notion of “*intersection* with a specific voxel” is well-defined. Early work in deep neural nets used these types of structural representations, though the volumetric approaches have been less prevalent recently for reasons that include: (i) size constraints (large proteins consume much memory, scaling with the cube of protein size, in terms of number of residues); (ii) mathematical considerations, such as this representation’s lack of rotational invariance (e.g., structures are manually rotated); (iii) fixed-grid volumetric models are inherently less flexible than graph representations (e.g. 3D images are static and cannot easily incorporate fluctuations, imparting a ‘brittleness’ to these types of data structures); and (iv) related to the issue of brittleness, there exists a rich and versatile family of graph-based algorithms, versus more limited (and less easily implemented) approaches for volumetric data.

A common approach to voxelize a protein structure into a dense grid is to calculate the distance of every atom to every voxel, then use the Lennard-Jones potential to map a scaled biophysical properties to each voxel [21, 22]. This works well for small proteins, but can take a long time for larger structures because of the $\mathcal{O}(n^2)$ runtime. A faster voxelization approach is to create a sparse grid, where only voxels that overlap a van der Waals volume around each atom. This can be done using kD-trees, which scale as $\mathcal{O}(n \log n)$ [12].

When treating proteins as 3D images, one must take into account the importance of rotational invariance. All structures must be rotated to achieve (ideally) uniform random sampling, which can be achieved via the 3D rotation group ($SO(3)$), formulated as a Haar distribution over unit quaternions; however, those numerical steps add significant computational overhead unless the model is already rotationally invariant, such as with equivariant neural networks [23].

2.2.4 Outline of this work

The remainder of this work presents Prop3D, a new protein domain structure dataset that includes (i) corrected/sanitized protein structures, (ii) biophysical properties for each atom and residue, to allow for multiple representation modes, as well as (iii) train, test & validation splits that have been specifically formulated for use in ML of proteins (to mitigate evolutionary data leakage).

2.3 Overview of the software & dataset

2.3.1 Architecture and design

The Prop3D-20sf dataset is created using two other frameworks we developed: (i) 'Meadowlark', for processing and interrogating individual protein structures and (ii) 'AtomicToil', for creation of massively parallel workflows of many thousands of structures. While each of these pieces of code are intricately woven together (in practice), giving Prop3D, it helps to consider them separately when examining their utility/capabilities and their respective roles in an overall Prop3D-based ML pipeline.

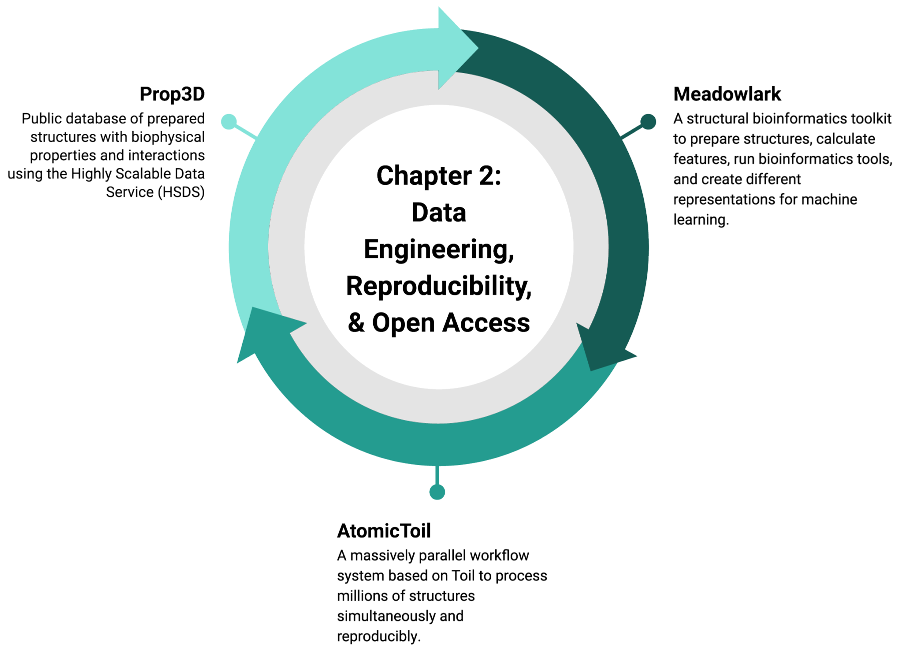

Prop3D
Public database of prepared structures with biophysical properties and interactions using the Highly Scalable Data Service (HSDS)
Meadowlark
A structural bioinformatics toolkit to prepare structures, calculate features, run bioinformatics tools, and create different representations for machine learning.
Chapter 2: Data Engineering, Reproducibility, & Open Access
AtomicToil
A massively parallel workflow system based on Toil to process millions of structures simultaneously and reproducibly.

Figure 2.3: Overview of Prop3D Components. We developed Prop3D as framework to create and share biophysical properties. We do this by creating two separate packages within the framework: (i) 'Meadowlark' to prepare structures, calculate features, and run **??**-ized tools; and (ii) 'AtomicToil' to run these calculations in parallel and on the cloud using Toil. The data from Prop3D is available as a publicly-available HSDS endpoint

2.3.2 Meadowlark: An extensible, Dockerized toolkit for reproducible, cross-platform structural bioinformatics workflows

In bioinformatics and computational biology more broadly, many tools and codes can be less than straightforward to install and operate locally: they each require various combinations of operating system configurations, specific versions of different languages and libraries (which may or may not be cross-compatible), have various interdependencies for installation/compilation (and for run-time execution), and so on. Moreover, considered across the community as a whole, researchers spend many hours installing (and perhaps even performance-tuning) these tools themselves, only to find that they are conducting similar development and upkeep of this computational infrastructure as are numerous other individuals; all the while, the data, results, and technical/methodological details underpinning the execution of a computational pipeline are typically never shared, at least not before the point of eventual publication (i.e., months to even years after the point at which it would have been most useful to others). Following the examples of the UC Santa Cruz Computational Genomics Laboratory (UCSC-CGL) and Global Alliance for Genomics & Health (GA4GH) [24], we Docker-ize common structural bioinformatics tools to make them easily deployable and executable on any machine, along with parsers to handle their outputs, all without leaving a top-level Python workflow. New software can be added into meadowlark if it exists as a Docker or Singularity container [?, 25]. For a list of codes and software tools that we have thus far made available, see Tables 2.3 & 2.4 or visit our Docker Hub.

2.3.3 AtomicToil: Mapping structural info to sets of massively parallel tasks

To construct and automate the deployment of massively parallel workflows in the cloud, we use a Python-based workflow management system (WMS) known as Toil [26]. Each top-level Toil job has child jobs and follow-on jobs, enabling the construction of complex MapReduce-like pipelines. A Toil workflow can be controlled locally, on the cloud (e.g., AWS, Kubernetes), or on a compute farm or a high-performance computing platform such as a Linux-based cluster (with a scheduler such as SLURM, Oracle Grid Engine, or the like). Background information on the data-flow paradigm, flow-based programming, task-oriented toolkits (like Toil), and related WMS concepts as they pertain to bioinformatics can be found in [27].

Within Prop3D, we specifically created multiple ways to instantiate a workflow:

1. Based on PDB files (can contain multiple chains or a single domain) to systematically map PDB files to jobs to run a given function.

2. Based on the CATH Hierarchy, where one job is created for each entry in the CATH hierarchy, with child jobs spawned for subsidiary levels in the hierarchy. Once the workflow reaches a job for each individual domain (or specified level), it will run a given, user-provisioned function.

New functions can be added into the workflow by creating new Toil job functions, which can be as simple as standalone Python functions with given, well-formed inputs.

2.3.4 Capabilities and Features

In this section we provide two examples of Prop3D usage, from relatively simple to more advanced: (i) protein structure preparation; and (ii) biophysical property calculations (and annotation).

Protein Structure Preparation

We 'clean' or 'sanitize' a starting protein structure by selecting the first model (from among multiple possible models in a PDB file), the correct chain, and the first alternate location (if multiple conformers are present for an atom/residue), and removing hetero-atoms (water or buffer molecules, other crystallization reagents, etc.); these steps are achieved via pdb-tools [28]. We then modify each domain structure via the following stages: (i) Build/model any missing residues with MODELLER [29]; (ii) Correct rotamers with SCWRL4 [30], e.g. if there are any missing atoms; and (3) Add hydrogens and perform a rough potential energy minimization with the PDB2PQR toolkit [31]. Note that this general workflow, schematized in Figure 2.4, was applied in constructing the Prop3D-20sf dataset.

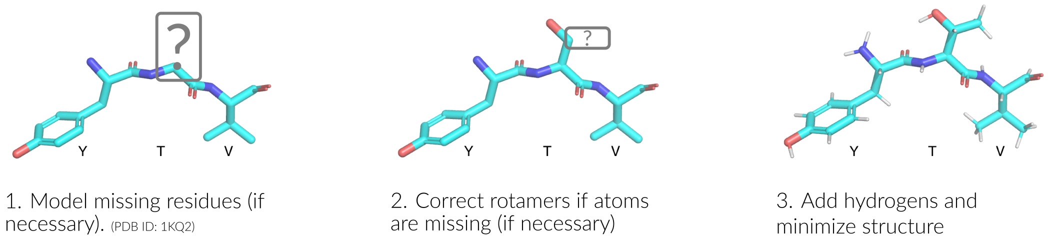

?
Y
T
V
1. Model missing residues (if necessary). (PDB ID: 1KQ2)
?
Y
T
V
2. Correct rotamers if atoms are missing (if necessary)
Y
T
V
3. Add hydrogens and minimize structure

Figure 2.4: Protein Preparation. Every domain is 'corrected' by adding missing atoms and residues, protonating, and energy minimizing the structure.

Biophysical Property Calculaton

The Prop3D toolkit enables one to rapidly and efficiently compute biophysical properties for all atoms and residues in a dataset of 3D structures (e.g., from the PDB or CATH).

For atom-level features, we create 'one-hot' encodings for 23 AutoDock atom names, 16 element names, and 21 amino acid residue types. We also include van der Waals radii, charges from PDB2PQR [31], electrostatic potentials computed via APBS [32], concavity measures from CX [33], various hydrophobicity features of the residue that an atom belongs to (Kyte-Doolite [34], Biological [35] and Octanol [36]), accessible surface area (per-atom, via FreeSASA [37], and per-residue via DSSP [38]). We also include different types of secondary structure information: 'one-hot' encodings for DSSP [38] 3- and 7- secondary structure classifications, as well as the backbone torsion angles (ϕ, ψ; along with embedded sine and cosine transformations of each). We also annotate aromaticity, and hydrogen bond acceptors and donors, based on AutoDock atom-name types. As a gauge of phylogenetic conservation, we include sequence entropy scores from EPPIC [39]. These biophysical, physicochemical, structural, and phylogenetic features are summarized in Figure 2.5 and are exhaustively enumerated in Table 2.6. Finally, we also provide functionality to create Boolean-valued descriptors from the corresponding continuous-valued quantities of a given feature via simple numerical thresholding (Table 2.7).

Some of the above properties are computed at the residue level and mapped to each atom in the residue (e.g., hydrophobicity is one such property); that is, the 'child' atom inherits the value of the given feature from its 'parent' residue. For other features, residue-level values are calculated by combining atomic quantities, via var-

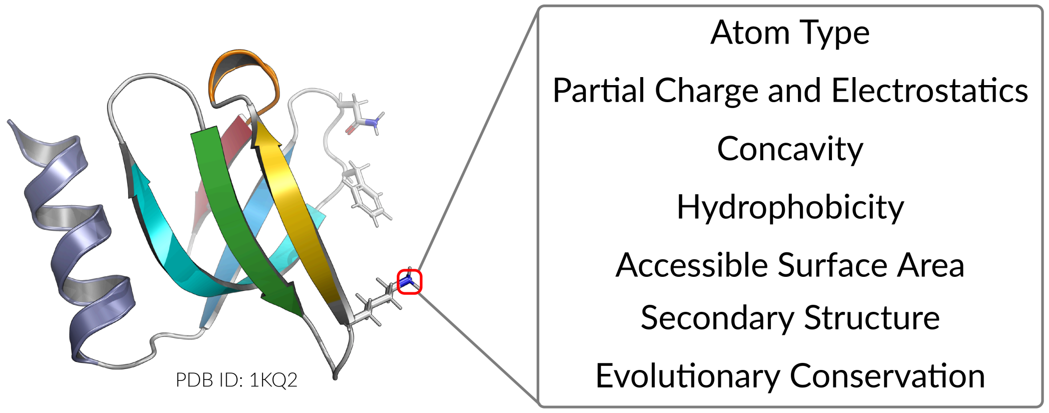


Figure 2.5: Biophysical Property Calculation. For each domain, we annotate every atom with following the features: atom type, element type, residue type, partial charge & electrostatics, concavity, hydrophobicity, accessible surface area, secondary structure type, and evolutionary conservation. For a full list of features, see Tables 2.6 & 2.7.

ious summation or averaging operations applied to the properties' numerical values (described in Table 2.6).

2.3.5 Dataset Design and Data Format

We employ the Hierarchical Data Format (HDF5 [40]), along with the Highly Scalable Data Service (HSDS), to handle the large amount of protein data in our massively parallel workflows. The HDF5 file format is a useful way to store and access large protein datasets because it allows us to chunk and compresses the CATH protein structure hierarchy in a scalable and efficient manner. Using this approach, instead of creating many individual files spread across multiple directories, we combine them into a 'single' file that is easily shareable and can be accessed via a hierarchical structure of groups and datasets, each with attached metadata. Moreover, the HSDS extension to this file-format system allows multiple readers and writers which, in combination with Toil, makes it extremely fast to create new datasets.

We note that many computational biologists are migrating to HDF5 [41, 42, 43] and HSDS [44] because it is fast to read binary data, easily shareable, and provides integrated metadata and other beneficial organization features (thus, e.g., facilitating data provenancing). Before HDF5 and HSDS and other binary formats came around, biological data formats for protein structure relied on human-readable ASCII files. Legacy PDB files have been the *de facto* format to store protein structures. Originally developed in 1976 to work with punch cards, legacy PDB files are ASCII files that have a fixed-column width and a max of 80 characters per line [45]. Only one biophysical property could be added into the B-factor column as a poorly designed workaround. Due to inflexibility of legacy PDB file format, the macromolecular Crystallographic Information Framework (mmCIF) file format was developed to be more structured and allowed for a plethora of biophysical properties [46]. Most recently due to the slow nature of reading ASCII files, the Macromolecular Transmission Format (MMTF) was developed to store protein structures in a compact binary format based on MessagePack format (version 5) [47, 48]. While the MMTF is almost ideal for machine learning, it still relies on individual files in a file system with no efficient, *distributed* mechanism to read in all files, no way to include metadata higher than residue level, and no ability to combine train/test splits directly into the schema, which were some of our motivating factors to use HDF5 and HSDS.

The Prop3D HDF5 file starts with the CATH database, which provides a hierarchical 'structure' that is naturally amenable to parallelization and efficient data traversal—namely, *Class* $\supset$ *Architecture* $\supset$ *Topology* $\supset$ *Homologous Superfamily*—as shown in Figure 2.6. A superfamily can be accessed by its CATH code as the group key (e.g., '2/60/40/10' for Immunoglobulin). Each superfamily is then split into two groups: '`domains`' (containing groups for each domain inside that superfamily) and '`data_splits`' (containing pre-computed train (80%), validation (10%), and test (10%) data splits for use in machine learning models, where each domain in each split is hard linked to the group for that domain). Each domain group contains datasets for different types of features: 'atoms', 'residues' and 'edges.' The 'Atoms' dataset contains information from the PDB ATOM field as well all of the biophysical properties calculated for each atom. 'Residues' contain biophysical properties of each residue and position (average of all of its atoms), for use in coarse-grained models. 'Edges' contains properties for each *residue* $\leftrightarrow$ *residue* interaction, thereby enabling the construction and annotation of contact maps.

In terms of the computational pipeline, HSDS allows HDF5 files to be stored in S3-like buckets, e.g. from AWS or MinIO, remotely and accessible via a REST API. HSDS data nodes and service nodes are controlled via a load balancer in Kubernetes too enable efficient, distributed mechanism to query of HDF5 file as well write data into the HDF5 also in a quick, efficient, distributed mechanism allowing. HSDS allows for multiple readers and multiple writers to read or write to the same file simultaneously, using the 'distributed' HDF5 multi-reader/multi-writer python library, h5pyd (See Fig. 2.7). We set up a local k3s instance, which is an easy-to-install lightweight distribution of Kubernetes that can run on a single machine along with MinIO S3 buckets.

When we created the Prop3D-20sf dataset, HSDS, in combination with a Toil-enabled workflow, allows for each parallel task to write to the same HDF5 file simultaneously. The Prop3D-20sf dataset can then be read in parallel as well, e.g. in PyTorch. We provide PyTorch Data Loaders to read the Prop3D dataset from the HSDS endpoint using multiple processes, available in our related DeepUrfold Python package [12]. When HSDS was used for training instead of raw ASCII files, we saw a speedup of 8 hours (from 24 hours to 16 hours of wallclock time) for training an immunoglobulin-specific variational autoencoder model with 25,524 featurized immunoglobulin domain structures (See Fig 2.8). Therefore, we found it clearly advantageous to utilize the parallelizable data-handler capacity provided by a system like HSDS.

2.4 Summary and Future Outlook

This work has presented 'Prop3D', a protein properties featurization and data-processing pipeline that we have developed and deployed. The Prop3D platform is extensible and scalable, can be used with local HPC resources as well as in the cloud, and allows one to systematically and reproducibly create comprehensive datasets using the Highly Scalable Data Service (HSDS). We have used Prop3D to create (and share) a new 'Prop3D-20sf' resource; this protein dataset, available as an HSDS endpoint, combines 3D coordinates with biophysical and evolutionary properties (for each atom), in each structural domain for the 20 most-highly populated homologous superfamilies (SF) in the CATH database. These 3D domains are sanitized via numerous steps, including clean-up of the covalent structure (e.g., adding missing atoms and residues) and physicochemical properties (protonation and energy minimization). Our database schema follows CATH's hierarchy, mapped to a system based on HDF5 files and including atomic-level features, residue-level features, residue-residue contacts, and pre-calculated train/test/validate splits (in ratios of 80/10/10) for each SF derived from CATH's sequence-identity–based clusters (e.g., 'S35' for groups of proteins culled at 35% sequence identity). We believe that Prop3D-20sf, and its underlying Prop3D framework, may be useful as a community resource in developing workflows that entail processing protein 3D structural information—particularly for pipelines that arise at the intersection of machine learning and structural bioinformatics.

This dataset can be used to compare sequence-based (1D), residue:residue graphs (2D), and structure-based (3D) methods. For example, one could train a supervised model with input being a protein sequence to predict a specific residue-based biophysical property. Similarly, unsupervised models can be trained using one or all of the biophysical properties to learn protein embeddings.

We built AtomicToil to enable the facile creation of reproducible workflows, starting with PDB files or by traversing the CATH hierarchy, as well as the Meadowlark toolkit to run Docker-ized structural bioinformatics software. While we primarily developed the tools described here in order to create the Prop3D-20sf dataset, we envision that the toolkit can be integrated into a feature-rich, standalone structural bioinformatics toolkit such as BioPython or Biotite.

2.5 Data Availability

Our code to run predefined workflows exists in our GitHub repository (`https://github.com/bouralab/Prop3D`) with scripts to set up HSDS and Kubernetes if running on your local system through k3s.

The pre-calculated features and data splits for 20 superfamlies exist in our HSDS endpoint at the University of Virginia (`hdf5://uvaarc01.virgnia.edu/bournelab/Prop3D.h5`) with the raw HDF5 on Zenodo (https://doi.org/10.5281/zenodo.6873024). These features can be read into a python program using h5pyd, our Prop3D library (https://github.com/bouralab/Prop3D), or through custom PyTorch data loaders available in our DeepUrfold (https://github.com/bouralab/DeepUrfold) GitHub repository. Finally, all of our Docker-ized tools can also be obtained from our Docker Hub at https://hub.docker.com/u/edraizen

We use Wikidata to cite the software we use as well create links to our code and data repositories (Q108040542).

Acknowledgements

We would like to thank Lane Rasberry for help editing the manuscript and providing support for Wikidata. We also thank Menuka Jaiswal, Saad Saleem, and Yonghyeon Kweon for early efforts on this project. Portions of this work were supported by the University of Virginia and NSF Career award MCB-1350957. EJD was supported by a University of Virginia Presidential Fellowship in Data Science.

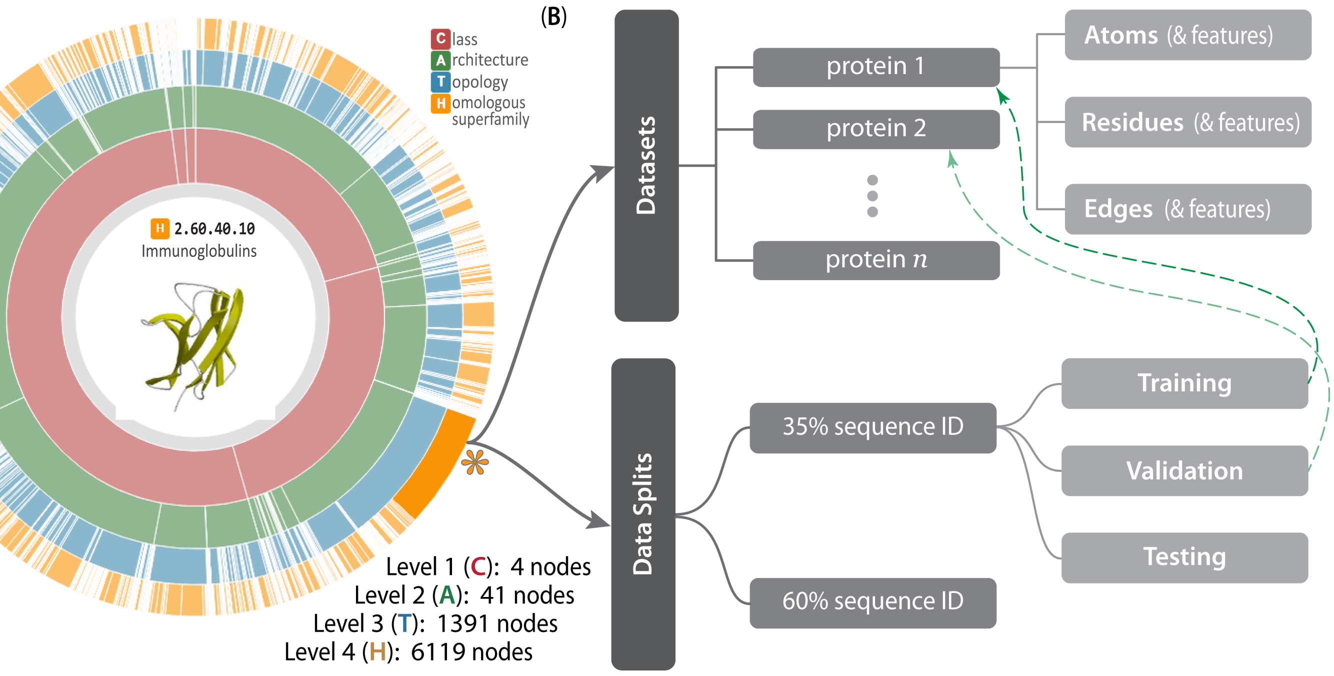


Figure 2.6: Hierarchical structure of Prop3D. The inherently hierarchical structure of CATH (A) is mirrored in the design schema underlying the Prop3D dataset (B), as illustrated here. Prop3D can be accessed as an HDF5 file seeded with the CATH hierarchy for all available superfamilies. For clarity, an example of one such superfamily is the individual H-group `2.60.40.10` (Immunoglobulins) shown here as the orange sector (denoted by an asterisk near 4 o'clock). Each such superfamily is further split into (i) the domain groups, with datasets provided for each domain (atomic features, residue features, and edge features), as delineated in the upper-half of (B), and (ii) pre-calculated data splits, shown in the lower-half of (B), which exist as hard-links (denoted as dashed green lines) to domain groups. (The 'sunburst' style CATH diagram, from cathdb.info, is under the Creative Commons Attribution 4.0 International License.

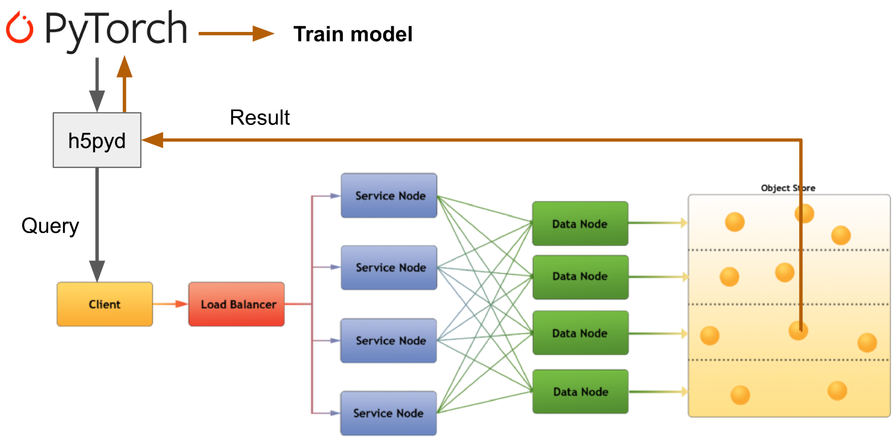
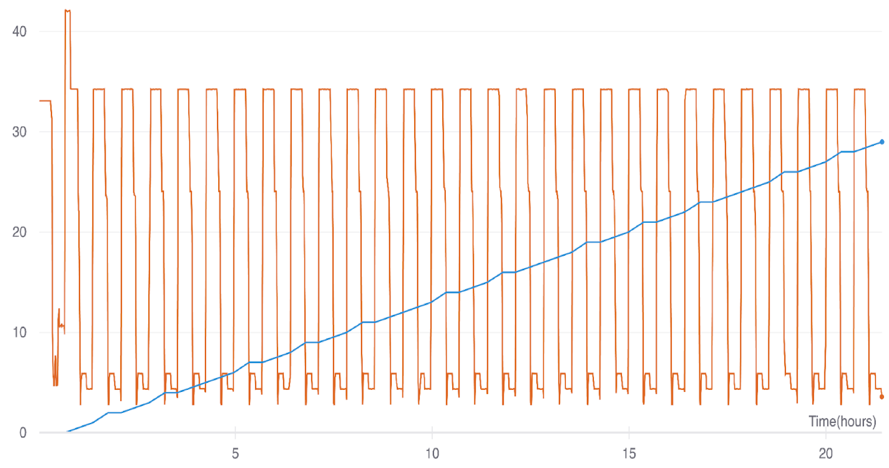


Figure 2.7: Cloud-based access to the Prop3D dataset with HSDS. HSDS creates Service Nodes, which are containers that handle query requests from the clients, and Data Nodes, which are containers that access the object storage in an efficient distributed manner. The Prop3D dataset can be used as input to train a machine learning model by either accessing the data through the python library h5pyd or though the DeepUrfold Python package that contains PyTorch data loaders [12]. accessed using Figure modified from the HSDS webpage available under an Apache 2.0 licenese (compatible with CC-by-4.0).

A. Before HSDS: PDB Files of atomic coordinates & CSV files of biophysical properties (All ASCII)



B. After HSDS: Combined atomic coordinates & biophysical properties in HDF format

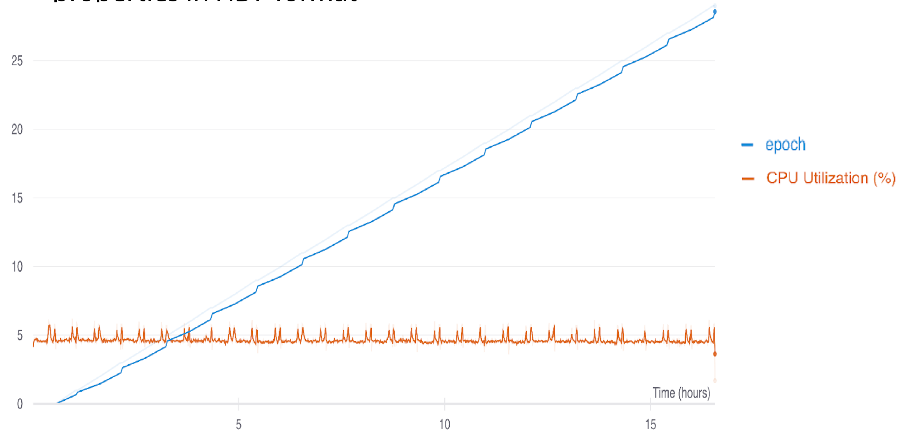


Figure 2.8: Improved training runtime when using HSDS. We trained an Immuniglobulic specific variational autoencoder with ≈25K domain structures using 64 cpus to process data and 4 gpus for 30 epochs [12]. **A.** Before we implemented HSDS, we stored domain structures as PDB files (parsed with BioPython) along with biophysical properties for all atoms in these PDB in separate PDB files as CSV files (parsed with Pandas). This took ≈24 hours to read ≈50K ASCII files. **B.** After we streamlined our process with HSDS, we improved our training runtime by ≈8 hours (total ≈16 hours) and more efficient CPU usage while reading all of the data. Images exported from our Weights and Biases training dashboard.

Bibliography

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

[2] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, Augustin Žídek, Tim Green, Kathryn Tunyasuvunakool, Stig Petersen, John Jumper, Ellen Clancy, Richard Green, Ankur Vora, Mira Lutfi, Michael Figurnov, Andrew Cowie, Nicole Hobbs, Pushmeet Kohli, Gerard Kleywegt, Ewan Birney, Demis Hassabis, and Sameer Velankar. AlphaFold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444, November 2021.

[3] S F Altschul, T L Madden, A A Schäffer, J Zhang, Z Zhang, W Miller, and D J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, Sep 1997.

[4] Yang Zhang and Jeffrey Skolnick. Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic Acids Research*, 33(7):2302–2309, Apr 2005.

[5] Sean Whalen, Jacob Schreiber, William S. Noble, and Katherine S. Pollard. Navigating the pitfalls of applying machine learning in genomics. *Nature Reviews Genetics*, 23(3):169–181, November 2021.

[6] Robbie P. Joosten, Fei Long, Garib N. Murshudov, and Anastassis Perrakis. The PDB_REDO server for macromolecular structure model optimization. *IUCrJ*, 1(4):213–220, May 2014.

[7] Peter Eastman, Jason Swails, John D. Chodera, Robert T. McGibbon, Yutong Zhao, Kyle A. Beauchamp, Lee-Ping Wang, Andrew C. Simmonett, Matthew P. Harrigan, Chaya D. Stern, Rafal P. Wiewiora, Bernard R. Brooks, and Vijay S. Pande. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLOS Computational Biology*, 13(7):e1005659, July 2017.

[8] Dan Graur and Wen-Hsiung Li. *Fundamentals of molecular evolution*. Oxford University Press, New York, NY, 2 edition, October 1999.

[9] Stephen K Burley, Charmi Bhikadiya, Chunxiao Bi, Sebastian Bittrich, Li Chen, Gregg V Crichlow, Cole H Christie, Kenneth Dalenberg, Luigi Di Costanzo, Jose M Duarte, Shuchismita Dutta, Zukang Feng, Sai Ganesan, David S Goodsell, Sutapa Ghosh, Rachel Kramer Green, Vladimir Guranović, Dmytro Guzenko, Brian P Hudson, Catherine L Lawson, Yuhe Liang, Robert Lowe, Harry Namkoong, Ezra Peisach, Irina Persikova, Chris Randle, Alexander Rose, Yana Rose, Andrej Sali, Joan Segura, Monica Sekharan, Chenghua Shao, Yi-Ping Tao, Maria Voigt, John D Westbrook, Jasmine Y Young, Christine Zardecki, and

Marina Zhuravleva. RCSB protein data bank: powerful new tools for exploring 3d structures of biological macromolecules for basic and applied research and education in fundamental biology, biomedicine, biotechnology, bioengineering and energy sciences. *Nucleic Acids Research*, 49(D1):D437–D451, November 2020.

[10] Adam J. Riesselman, John B. Ingraham, and Debora S. Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature Methods*, 15(10):816–822, September 2018.

[11] Ian Walsh, Gianluca Pollastri, and Silvio C E Tosatto. Correct machine learning on protein sequences: a peer-reviewing perspective. *Brief. Bioinform.*, 17(5):831–840, September 2016.

[12] Eli J. Draizen, Stella Veretnik, Cameron Mura, and Philip E. Bourne. Deep generative models of protein structure uncover distant relationships across a continuous fold space. *bioRxiv*, 2022.

[13] The UniProt Consortium. UniProt: The universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, November 2020.

[14] Ian Sillitoe, Nicola Bordin, Natalie Dawson, Vaishali P Waman, Paul Ashford, Harry M Scholes, Camilla S M Pang, Laurel Woodridge, Clemens Rauer, Neeladri Sen, Mahnaz Abbasian, Sean Le Cornu, Su Datt Lam, Karel Berka, Ivana Hutařová Varekova, Radka Svobodova, Jon Lees, and Christine A Orengo. CATH: increased structural coverage of functional space. *Nucleic Acids Research*, 49(D1):D266–D273, November 2020.

[15] Inbal Halperin, Dariya S Glazer, Shirley Wu, and Russ B Altman. The FEATURE framework for protein function annotation: modeling new functions, im-

proving performance, and extending to novel applications. *BMC Genomics*, 9(S2), September 2008.

[16] Michael Bernhofer, Christian Dallago, Tim Karl, Venkata Satagopam, Michael Heinzinger, Maria Littmann, Tobias Olenyi, Jiajun Qiu, Konstantin Schütze, Guy Yachdav, Haim Ashkenazy, Nir Ben-Tal, Yana Bromberg, Tatyana Goldberg, Laszlo Kajan, Sean O'Donoghue, Chris Sander, Andrea Schafferhans, Avner Schlessinger, Gerrit Vriend, Milot Mirdita, Piotr Gawron, Wei Gu, Yohan Jarosz, Christophe Trefois, Martin Steinegger, Reinhard Schneider, and Burkhard Rost. PredictProtein - Predicting Protein Structure and Function for 29 Years. *Nucleic Acids Research*, 49(W1):W535–W540, 05 2021.

[17] Bi Zhao, Akila Katuwawala, Christopher J Oldfield, A Keith Dunker, Eshel Faraggi, Jörg Gsponer, Andrzej Kloczkowski, Nawar Malhis, Milot Mirdita, Zoran Obradovic, Johannes Söding, Martin Steinegger, Yaoqi Zhou, and Lukasz Kurgan. DescribePROT: database of amino acid-level protein structure and function predictions. *Nucleic Acids Research*, 49(D1):D298–D308, October 2020.

[18] Raphael J. L. Townshend, Martin Vögele, Patricia Suriana, Alexander Derry, Alexander Powers, Yianni Laloudakis, Sidhika Balachandar, Bowen Jing, Brandon Anderson, Stephan Eismann, Risi Kondor, Russ B. Altman, and Ron O. Dror. Atom3d: Tasks on molecules in three dimensions. *arXiv*, 2020.

[19] Mohammed AlQuraishi. ProteinNet: a standardized data set for machine learning of protein structure. *BMC Bioinformatics*, 20(1), June 2019.

[20] Jonathan E. King and David Ryan Koes. Sidechainnet: An all-atom protein structure dataset for machine learning, 2020.

[21] J Jiménez, S Doerr, G Martínez-Rosell, A S Rose, and G De Fabritiis. DeepSite: protein-binding site predictor using 3d-convolutional neural networks. *Bioinformatics*, 33(19):3036–3042, May 2017.

[22] Martin Simonovsky and Joshua Meyers. DeeplyTough: Learning structural comparison of protein binding sites. *Journal of Chemical Information and Modeling*, 60(4):2356–2366, February 2020.

[23] Fabian B. Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks. *CoRR*, abs/2006.10503, 2020.

[24] Denis Yuen, Louise Cabansay, Andrew Duncan, Gary Luu, Gregory Hogue, Charles Overbeck, Natalie Perez, Walt Shands, David Steinberg, Chaz Reid, Nneka Olunwa, Richard Hansen, Elizabeth Sheets, Ash O'Farrell, Kim Cullion, Brian D O'Connor, Benedict Paten, and Lincoln Stein. The dockstore: enhancing a community platform for sharing reproducible and accessible computational protocols. *Nucleic Acids Research*, 49(W1):W624–W632, May 2021.

[25] Gregory M. Kurtzer, Vanessa Sochat, and Michael W. Bauer. Singularity: Scientific containers for mobility of compute. *PLOS ONE*, 12(5):e0177459, May 2017.

[26] John Vivian, Arjun Arkal Rao, Frank Austin Nothaft, Christopher Ketchum, Joel Armstrong, Adam Novak, Jacob Pfeil, Jake Narkizian, Alden D Deran, Audrey Musselman-Brown, Hannes Schmidt, Peter Amstutz, Brian Craft, Mary Goldman, Kate Rosenbloom, Melissa Cline, Brian O'Connor, Megan Hanna, Chet Birger, W James Kent, David A Patterson, Anthony D Joseph, Jingchun Zhu, Sasha Zaranek, Gad Getz, David Haussler, and Benedict Paten. Toil enables

reproducible, open source, big biomedical data analyses. *Nature Biotechnology*, 35(4):314–316, April 2017.

[27] Marcin Cieślik and Cameron Mura. A lightweight, flow-based toolkit for parallel and distributed bioinformatics pipelines. *BMC Bioinformatics*, 12:61, February 2011.

[28] JPGLM Rodrigues, JMC Teixeira, M Trellet, and AMJJ Bonvin. pdb-tools: a swiss army knife for molecular structures. *F1000Research*, 7(1961), 2018.

[29] Benjamin Webb and Andrej Sali. Comparative protein structure modeling using MODELLER. *Current Protocols in Bioinformatics*, 54(1), June 2016.

[30] Georgii G. Krivov, Maxim V. Shapovalov, and Roland L. Dunbrack. Improved prediction of protein side-chain conformations with SCWRL4. *Proteins: Structure, Function, and Bioinformatics*, 77(4):778–795, May 2009.

[31] T. J. Dolinsky, P. Czodrowski, H. Li, J. E. Nielsen, J. H. Jensen, G. Klebe, and N. A. Baker. PDB2pqr: expanding and upgrading automated preparation of biomolecular structures for molecular simulations. *Nucleic Acids Research*, 35(Web Server):W522–W525, May 2007.

[32] Elizabeth Jurrus, Dave Engel, Keith Star, Kyle Monson, Juan Brandi, Lisa E. Felberg, David H. Brookes, Leighton Wilson, Jiahui Chen, Karina Liles, Minju Chun, Peter Li, David W. Gohara, Todd Dolinsky, Robert Konecny, David R. Koes, Jens Erik Nielsen, Teresa Head-Gordon, Weihua Geng, Robert Krasny, Guo-Wei Wei, Michael J. Holst, J. Andrew McCammon, and Nathan A. Baker. Improvements to the scpAPBS/scp biomolecular solvation software suite. *Protein Science*, 27(1):112–128, October 2017.

[33] A. Pintar, O. Carugo, and S. Pongor. CX, an algorithm that identifies protruding atoms in proteins. *Bioinformatics*, 18(7):980–984, July 2002.

[34] Jack Kyte and Russell F. Doolittle. A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology*, 157(1):105–132, May 1982.

[35] Tara Hessa, Hyun Kim, Karl Bihlmaier, Carolina Lundin, Jorrit Boekel, Helena Andersson, IngMarie Nilsson, Stephen H. White, and Gunnar von Heijne. Recognition of transmembrane helices by the endoplasmic reticulum translocon. *Nature*, 433(7024):377–381, January 2005.

[36] William C. Wimley and Stephen H. White. Experimentally determined hydrophobicity scale for proteins at membrane interfaces. *Nature Structural &amp Molecular Biology*, 3(10):842–848, October 1996.

[37] Simon Mitternacht. FreeSASA: An open source c library for solvent accessible surface area calculations. *F1000Research*, 5:189, February 2016.

[38] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, December 1983.

[39] Spencer Bliven, Aleix Lafita, Althea Parker, Guido Capitani, and Jose M. Duarte. Automated evaluation of quaternary structures from protein crystals. *PLOS Computational Biology*, 14(4):e1006104, April 2018.

[40] The HDF Group. Hierarchical Data Format, version 5, 1997-NNNN. https://www.hdfgroup.org/HDF5/.

[41] Bilal Shaikh, Gnaneswara Marupilla, Mike Wilson, Michael L Blinov, Ion I Moraru, and Jonathan R Karr. RunBioSimulations: an extensible web application that simulates a wide range of computational modeling frameworks, algorithms, and formats. *Nucleic Acids Research*, 49(W1):W597–W602, May 2021.

[42] Nicolas Renaud, Cunliang Geng, Sonja Georgievska, Francesco Ambrosetti, Lars Ridder, Dario F. Marzella, Manon F. Réau, Alexandre M. J. J. Bonvin, and Li C. Xue. DeepRank: a deep learning framework for data mining 3d protein-protein interfaces. *Nature Communications*, 12(1), December 2021.

[43] M. Réau, N. Renaud, L. C. Xue, and A. M. J. J. Bonvin. Deeprank-gnn: A graph neural network framework to learn patterns in protein-protein interfaces. *bioRxiv*, dec 2021.

[44] Andrew Freiburger, Bilal Shaikh, and Jonathan Karr. Biosimulations: a platform for sharing and reusing biological simulations, Feb 2022.

[45] Helen M. Berman. The protein data bank: a historical perspective. *Acta Crystallographica Section A Foundations of Crystallography*, 64(1):88–95, December 2007.

[46] Philip E. Bourne, Helen M. Berman, Brian McMahon, Keith D. Watenpaugh, John D. Westbrook, and Paula M.D. Fitzgerald. [30] macromolecular crystallographic information file. In *Methods in Enzymology*, pages 571–590. Elsevier, 1997.

[47] Anthony R. Bradley, Alexander S. Rose, Antonín Pavelka, Yana Valasatava, Jose M. Duarte, Andreas Prlić, and Peter W. Rose. MMTF—an efficient file format for the transmission, visualization, and analysis of macromolecular structures. *PLOS Computational Biology*, 13(6):e1005575, June 2017.

[48] Yana Valasatava, Anthony R. Bradley, Alexander S. Rose, Jose M. Duarte, Andreas Prlić, and Peter W. Rose. Towards an efficient compression of 3d coordinates of macromolecular structures. *PLOS ONE*, 12(3):e0174846, March 2017.

[49] A. Bondi. van der waals volumes and radii. *The Journal of Physical Chemistry*, 68(3):441–451, March 1964.

Supplemental Information

Name	Description	Wikidata entry
BLAST	search sequences (or groups of sequences) against the NR database or custom database	Q286820
DeepSequence	A generative latent variable model for biological sequence families	Q114841036
ESM	Pretrained language models for proteins	Q114841163
EVcouplings	Evolutionary couplings from protein and RNA sequence alignments	Q114841016
HMMER	build hmmer models with a group of sequences or at a given level of the CATH hierarchy, search a a group of sequences with a pretrained model, or use jackhmmer starting from a single sequence	Q5631078
MMSeqs2	ultra fast and sensitive search and clustering suite	Q114840759
MUSCLE	Multiple Sequence Aligner	Q6719088
SeqDesign	Protein design and variant prediction using autoregressive generative models	Q114841058
USEARCH	High-throughput sequence search and clustering analysis tool	Q114841186

Table 2.3: Sequence-based bioinformatics tools available in Prop3D. Most of these tools have been dockerized, available at our Docker Hub (https://hub.docker.com/u/edraizen)

Name	**Description/purpose** (in this context)	Wikidata entry
APBS	Adaptive Poisson-Boltzmann Solver, used here to calculate the electrostatic potential for each atom in a given protein	Q65072984
Consurf	Get pre-calculated conservation scores	Q112888886
CNS	Energy minimize a given structure	Q5191443
CX	Get curvature for each atom in a given protein	Q114841750
DSSP	calculate secondary structure and accessibility for each residue in a given structure	Q5206192
EPPIC	Calculate sequence conservation scores for a given protein and obtain biologically relevant protein interactions (i.e., not resulting from crystal packing)	Q114841783
foldseek	Fast searching and clustering of protein structure databases	Q114840749
FreeSASA	Get solvent accessibility of each atom in a given protein	Q114841793
Geometricus	A structure-based, alignment-free embedding approach for proteins, utilizing moment invariants	Q114840743
HADDOCK	Dock two proteins or refine the conformation of two docked proteins	Q114841798
MaxCluster	Cluster very similar structures	Q114840623
MGLTools	Convert atom names to Autodock names and PDBQT	Q114840701
MM-Align	Align two protein complexes	Q114841843
mTM-Align	Multiple structure alignment	Q114841813
MODELLER	Create full atom structures from C_α only models, mutate structures with different amino acids, 'remodel structure' to energy minimize, and model loops	Q3859815
MSMS	Calculate molecular surfaces and create meshes	Q114841806
Multivalue	Merge electrostatic values from multiple atoms. on a protein surface	Q114840933
OpenBabel	Convert to PDBQT format for AutoDock atom naming	Q612752
PDB2PQR	Protonate, debump hydrogens, and standardise naming in a given protein	Q62856803
pdb-tools	Swiss army knife of tools to manipulate PDB files	Q114840802
PRODIGY	Predict binding affinities (Kd) and Fraction of Common contacts in complexes	Q114840854
REDUCE	Protonate and de-protonate structures	Q114840896
SCWRL4	Correct side-chains using the Dunbrack rotamer library	Q114840881
TM-Align	Align two or more protein 3D structures	Q114840775

Table 2.4: Structural bioinformatics software available in Prop3D. Most of these tools have been dockerized, available at our Docker Hub (https://hub.docker.com/u/edraizen)

Name	Description/purpose (in this context)	Wikidata entry
AlphaFold2	Deep learning-based code for high-accuracy protein 3D structure prediction	Q107711739
AutoDock	A suite of automated protein docking tools	Q4826062
AWS	Amazon Web Services, on-demand cloud computing platforms	Q456157
BioPython	General-purpose collection of open-source tools for computational biology	Q4118434
Biotite	A comprehensive library for computational molecular biology	Q114859551
Docker	Open-source software for deploying containerized applications	Q15206305
HDF5	Hierarchical Data Format, version 5	Q1069215
HSDS	Cloud-native, service based access to HDF data	Q114859023
h5pyd	Python client library for HDF5 REST interface	Q114859536
Kubernetes	Software to manage containers on a server-cluster	Q22661306
k3s	A light-weight Kubernetes distribution for small servers	Q114860267
MinIO	Cloud storage server compatible with Amazon S3	Q28956397
NumPy	Numerical programming package for the Python programming language	Q197520
Pandas	Python library for data manipulation and analysis	Q15967387
PyTorch	Open-source, Python-based machine learning library	Q47509047
Toil	Enables creation and deployment of massively parallel workflows in Python	Q114858329
Singularity	Open-source container software for scientific environments	Q51294208
SLURM	Free and open-source job scheduler for Linux and similar computers to create a compute cluster	Q3459703
Oracle Grid Engine	Supercomputer batch-queuing system	Q2708256
Weights and Biases (wandb)	Python library to track machine learning experiments, version data and manage models	Q107382092

Table 2.5: Other mentioned software.

Feature	Voxel Aggregation Rule	Residue Aggregation Rule	Source Software or Database	Wikidata entry
H	max		MGLTools	Q114840701
HD	max		MGLTools	Q114840701
HS	max		MGLTools	Q114840701
C	max		MGLTools	Q114840701
A	max		MGLTools	Q114840701
N	max		MGLTools	Q114840701
NA	max		MGLTools	Q114840701
NS	max		MGLTools	Q114840701
OA	max		MGLTools	Q114840701
OS	max		MGLTools	Q114840701
F	max		MGLTools	Q114840701
MG	max		MGLTools	Q114840701
P	max		MGLTools	Q114840701
SA	max		MGLTools	Q114840701
S	max		MGLTools	Q114840701
CL	max		MGLTools	Q114840701
CA	max		MGLTools	Q114840701
MN	max		MGLTools	Q114840701
FE	max		MGLTools	Q114840701
ZN	max		MGLTools	Q114840701
BR	max		MGLTools	Q114840701
I	max		MGLTools	Q114840701
Unk_atom	max		MGLTools	Q114840701
C_elem	max		PDB File	
N_elem	max		PDB File	
O_elem	max		PDB File	

Continued on next page

Feature	Voxel Aggregation Rule	Residue Aggregation Rule	Source Software or Database	Wikidata entry
S_elem	max		PDB File	
H_elem	max		PDB File	
F_elem	max		PDB File	
MG_elem	max		PDB File	
P_elem	max		PDB File	
CL_elem	max		PDB File	
CA_elem	max		PDB File	
MN_elem	max		PDB File	
FE_elem	max		PDB File	
ZN_elem	max		PDB File	
BR_elem	max		PDB File	
I_elem	max		PDB File	
Unk_elem	max		PDB File	
vdw	mean	✓	[49]	
partial charge (charge)	mean	sum	Pdb2Pqr	Q62856803
electrostatic_potential	mean	sum	APBS	Q65072984
concavity (cx)	mean	mean	CX	Q114841750
hydrophobicity	mean	✓	Kyte-Doolittle [34]	
biological_hydrophobicity	mean	✓	[35]	
octanal_hydrophobicity	mean	✓	Wimley-White [36]	
atom_asa	mean		FreeSASA	Q114841793
residue_rasa	mean	✓	DSSP	Q5206192
ALA	max	✓	PDB File	
CYS	max	✓	PDB File	
ASP	max	✓	PDB File	
GLU	max	✓	PDB File	
PHE	max	✓	PDB File	
GLY	max	✓	PDB File	
HIS	max	✓	PDB File	
ILE	max	✓	PDB File	
LYS	max	✓	PDB File	
LEU	max	✓	PDB File	
MET	max	✓	PDB File	
ASN	max	✓	PDB File	
PRO	max	✓	PDB File	
GLN	max	✓	PDB File	
ARG	max	✓	PDB File	
SER	max	✓	PDB File	
THR	max	✓	PDB File	
VAL	max	✓	PDB File	
TRP	max	✓	PDB File	
TYR	max	✓	PDB File	
Unk_residue	max	✓	PDB File	
phi	mean	✓	BioPython	Q4118434
phi_sin	mean	✓	NumPy	
phi_cos	mean	✓	NumPy	
psi	mean	✓	BioPython	Q4118434.

Continued on next page

Feature	Voxel Aggregation Rule	Residue Aggregation Rule	Source Database	Software or	Wikidata entry
psi_sin	mean	✓	NumPy		
psi_cos	mean	✓	NumPy		
is_helix	max	✓	DSSP		Q5206192
is_sheet	max	✓	DSSP		Q5206192
Unk_SS	max	✓	DSSP		Q5206192
is_regular_helix	max	✓	DSSP		Q5206192
is_beta_bridge	max	✓	DSSP		Q5206192
is_extended_strand	max	✓	DSSP		Q5206192
is_310_helix	max	✓	DSSP		Q5206192
is_pi_helix	max	✓	DSSP		Q5206192
is_hbond_turn	max	✓	DSSP		Q5206192
is_bend	max	✓	DSSP		Q5206192
no_ss	max	✓	DSSP		Q5206192
hydrophobic_atom	max		MGLTools		Q114840701
aromatic_atom	max		MGLTools		Q114840701
hbond_acceptor	max		MGLTools		Q114840701
hbond_donor	max		MGLTools		Q114840701
metal	max		MGLTools		Q114840701
eppic_entropy	min	✓	EPPIC		Q114841783

Table 2.6: All calculated and extracted features. Voxel aggregation method is used to combine two or more atom features if they occupy the same voxels after van der walls sphere volume voxelation. Residue Aggregation Rule is how the feature is aggregated from atom to residue if present in the residue feature. A ✓ indicates if the feature was calculated at the residue level and mapped down to the atom level.

Continued on next page

Boolean Feature	Source Feature	Equality	Threshold
neg_charge	charge	$<$	0.
pos_charge	charge	$>$	0
is_electronegative	electrostatic_potential	$<$	0.
is_concave	cx	$\leq$	2
is_hydrophobic	hydrophobicity	$>$	0
residue_buried	residue_rasa	$<$	0.2
is_conserved	eppic_entropy	$<$	0.5

Table 2.7: Boolean Features converted from continuous values.

Chapter 3

DeepUrfold: Deep Generative Models of Protein Structure Uncover Distant Relationships Across a Continuous Fold Space

Eli J. Draizen[1,2*], Stella Veretnik[2], Cameron Mura[1,2*], and Philip E. Bourne[1,2]

[1]Department of Biomedical Engineering, University of Virginia, Charlottesville, VA, USA

[2]School of Data Science, University of Virginia, Charlottesville, VA, USA

[*]To whom correspondence should be addressed.

Available on bioRxiv as https://doi.org/10.1101/2022.07.29.501943. Will submit to Bioinformatics.

Abstract

Motivation: Our views of fold space implicitly rest upon many assumptions that impact how we analyze, interpret and understand biological systems—from protein structure comparison and classification to function prediction and evolutionary analyses. For instance, is there an optimal granularity at which to view protein structural similarities (e.g., architecture, topology or some other level)? Similarly, the discrete/continuous dichotomy of fold space is central in structural bioinformatics, but remains unresolved. Discrete views of fold space bin 'similar' folds into distinct, non-overlapping groups; unfortunately, such binning may inherently miss many remote relationships. While hierarchical databases like CATH, SCOP and ECOD represent major steps forward in protein classification, a scalable, objective and conceptually flexible method, with less reliance on assumptions and heuristics, could enable a more systematic and nuanced exploration of fold space, particularly as regards evolutionarily-distant relationships.

Results: Building upon a recent 'Urfold' model of protein structure, we have developed a new approach to analyze protein structure relationships. Termed 'DeepUrfold', the method is rooted in deep generative modeling, and we find it to be useful for comparative analysis across the protein universe. Critically, DeepUrfold leverages its deep generative model's embeddings, which represent a distilled, lower-dimensional space of a given protein and its amalgamation of sequence, structure and biophysical properties. Notably, DeepUrfold is structure-*guided*, versus being purely structure-based, and its architecture allows each trained model to learn protein features (structural and otherwise) that, in a sense, 'define' different superfamilies. Deploying DeepUrfold with CATH suggests a new, mostly-continuous view of fold space—a view that extends beyond simple 3D structural/geometric similarity, towards the realm of integrated *sequence↔structure↔function* properties. We find that such an approach can quantitatively represent and detect evolutionarily-remote relationships that evade existing methods.

Availability: Our results can be explored at https://bournelab.org/research/DeepUrfold/; code is available at http://www.github.com/bouralab/DeepUrfold and data are at https://doi.org/10.5281/zenodo.6916524.

Contact: e.draizen@gmail.com and cmura@virginia.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

3.1 Introduction

The precise historical trajectory of the protein universe [1] remains quite murky, and likely corresponds to an evolution from (proto-)peptides, to protein domains, to multi-domain proteins [2]. Presumably, the protein universe—by which we mean the set of all proteins (known or unknown, ancestral or extent)—did not spontaneously arise with intact, full-sized domains. Rather, smaller, sub-domain–sized protein fragments likely preceded more modern domains; the genomic elements encoding these primitive fragments were subject to natural evolutionary processes of duplication, mutation and recombination to give rise to extant domains found in contemporary proteins [2, 3, 4, 5, 6]. Our ability to detect common polypeptide fragments, shared amongst at least two domains (in terms of either sequence or structure), relies upon (i) having an accurate similarity metric and (ii) a suitable random/background distribution (i.e., null model) for distances under this metric; historically, such metrics have been rooted in the comparison of either amino acid sequences or three-dimensional (3D) structures, often for purposes of exploring protein fold space.

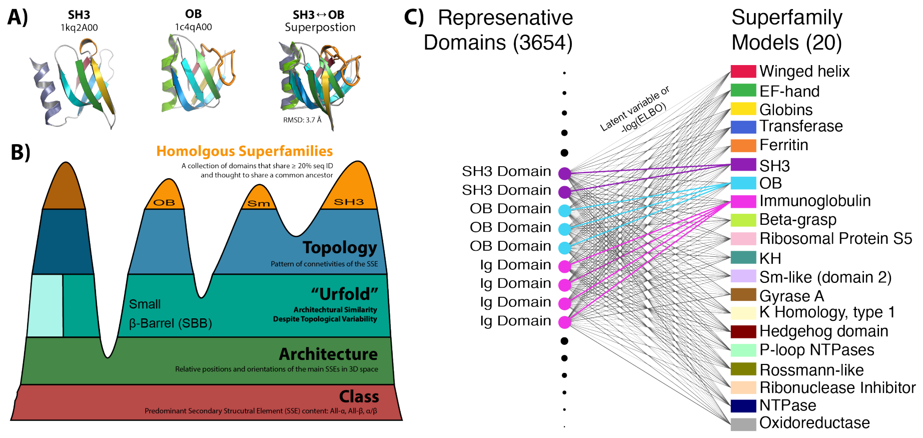


Figure 3.1: Overview of the Urfold model and DeepUrfold approach to identify domains that might reflect the phenomenon of 'architectural similarity despite topological variability.' (A) The SH3 and OB domains are prototypical members of the small β-barrel (SBB) urfold because they have the same barrel architecture, yet different strand topologies: they have strikingly similar 3D structures and share extensive functional similarities (e.g. PPI binding on the same edge-strand, involvement in nucleic acid–binding and processing pathways [7, 8]), yet these similarities are obscured by the SH3 and OB superfolds having been classified differently. In the case of the SBB urfold, the loops linking the strands are permuted in the SH3 and OB, yielding the different topologies seen in their 3D superposition. (B) If the Urfold phenomenon is viewed in terms of CATH, it is hypothesized to be a discrete structural entity that lies between the Architecture and Topology strata, as schematized here. (C) DeepUrfold, which applies deep learning to the Urfold conceptualization of protein structure, identifies new potential urfolds by creating 20 SF-specific VAE neural network models and comparing output scores from all representative domains from those superfamilies (numbering 3,654) to every other SF model. As the first-computed metric, we can imagine comparing the latent variables from domain representatives using models trained on the same SF (colored lines; see Fig. 3.3); then, we also perform an all-vs-all comparison to begin mapping fold space, which we (non-hierarchically) cluster via stochastic block models (SBMs; Fig. 3.4).

3.1.1 Fold Space, Structural Transitions & Fragments

Fold space[1], as the collection of all unique protein folds, is a many-to-one mapping: vast swaths of sequence space map to fold $\mathcal{A}$, another vast swath maps to fold $\mathcal{B}$, a narrower range might map to fold $\mathcal{C}$, and so on. Two proteins that are closely related (evolutionarily) might adopt quite similar folds ($\mathcal{A}$, $\mathcal{A}'$), leading to their proximity in this high-dimensional space. Traditionally, fold space has been examined by hierarchically clustering domains based upon 3D structure comparison; in such approaches, whatever metric is used for the comparison can be viewed as structuring the space. The transition of a protein sequence from one fold to another, whether it be nearby ($\mathcal{A} \to \mathcal{A}'$) or more distant ($\mathcal{A} \to \mathcal{B}$), and be it naturally (via evolution) or artificially (via design/engineering), likely occurs over multiple intermediate steps. These mechanistic steps include processes such as combining or permuting short secondary structural segments or longer segments (such as whole secondary structural elements [SSEs]), or mutating individual residues via nonsynonymous substitutions [10, 11, 12, 13, 5]. In general, each such step may yield a new 3D structure, and that structure may correspond to the same or a different fold. Similarities across these transitional states blur the boundaries that delineate distinct groups—increasing or decreasing a relatively arbitrary and heuristic quantity (such as an RMSD or other similarity threshold) can change which structures belong to which groups. In this sense, the discrete versus continuous duality of fold space can be viewed largely as a matter of semantics or thresholding, versus any 'real' (intrinsic or fundamental) feature of the space itself [14].

Despite their limitations, it was pairwise similarity metrics in structure space that

[1] The term "protein structure space" means the set of all protein 3D structures, known and unknown; the term "fold space" refers to the set of all protein folds. Though not strictly equivalent [9], we treat these terms interchangeably here.

first indicated remote connections in a continuous fold space via shared fragments [15] and references therein). In an early landmark study, [16] created an all-by-all similarity matrix from 3D structural alignments and discovered that the protein universe harbors five peptide 'attractors', representing frequently-adopted folding motifs (e.g., the β-meander). Later, similar pairwise analyses across protein structure space showed that 'all-α' and 'all-β' proteins are separated by 'α/β' proteins [17]. All-by-all similarity metrics for full domains (or fragments thereof) can be equivalently viewed as a graph-theoretic adjacency matrix, thus enabling the creation of a network representation of fold space. Such networks have been found to be nearly connected, linking domains in 4-8 hops [18, 19, 20].

Graph-based representations of individual proteins have also motivated the study of common short (sub-domain) fragments. In pioneering studies, [21, 22] found maximal common cliques of connected SSEs in a graph-based protein representation; their model took SSEs (helices, strands) as vertices and mapped the pairwise geometric relationships between SSEs (distances, angles, etc.) to the graph's edges. In that work, 80% of folds shared common cliques with other folds, and these were quantified by a new term called 'gregariousness'.

Although short, sub-domain–sized peptide fragments have been thoroughly studied, relatively few approaches have taken an evolutionary perspective, in the context of a continuous fold space. [23] identified common loop fragments flanked by SSEs, called Elementary Functional Loops (EFL), that couple in 3D space to perform enzymatic activity. [6] noticed that peptide fragments, called 'protodomains', are often composed (with C_2 internal symmetry) to give a larger, full-sized domain. Most recently, [4] identified common fragments between metal-binding proteins using *'sahle'*, a new length-dependent structural alignment similarity metric.

The two state-of-the-art, evolution-based fragment libraries that are currently available, namely ‘primordial peptides’ [2] and ‘themes’ [24], involved creation of a set of common short peptide fragments based on HHsearch [25] profiles for proteins in SCOP and ECOD, respectively. The sizes of the libraries created by these two sequence-driven approaches (40 primordial peptides, 2195 themes) vary greatly, reflecting different stringencies of thresholds (and, ultimately, their different goals).

Another approach to study shared, commonly-occurring fragments is to represent a protein domain as a vector of fragments. For example, the *FragBag* method [26] describes a protein by the occurrence of fragments in a clustered fragment library [27]. A recent and rather unique approach, *Geometricus* [28], creates protein embeddings by taking two parallel approaches to fragmentation: (i) a k-mer based fragmentation runs along the sequence (yielding contiguous segments), while (ii) a radius-based fragmentation uses the method of spatial moment invariants to compute (potentially non-contiguous) geometric ‘fragments’ for each residue position and its neighborhood within a given radius, which are then mapped to ‘shape-mers’. Conceptually, this allowance for discontinuous fragments is a key step in allowing an algorithm to bridge more of fold space, as similarities between such non-contiguous fragments can imply an ancestral (contiguous) polypeptide that duplicated and lost one or more N- or C-terminal SSEs in a ”creative destruction” process that yields two different folds (i.e., different topologies) but similar architectures [13, 5].

3.1.2 Limitations of Hierarchical Systems, The Urfold

The conventional view of fold space as the constellation of all folds, grouped by their similarities to one another, largely rests upon pioneering work in hierarchically clustering domains based upon 3D structure comparison, as exemplified in databases such as CATH [29], SCOP [30, 31], and ECOD [32]. Despite being some of the most comprehensive resources available in protein science, these databases have intrinsic limitations that stem from their fundamental structuring scheme, reflecting assumptions and constraints of any hierarchical system (e.g., assigning a given protein sequence to one mutually exclusive bin versus others); in this design schema, domains with the same fold or superfamily (SF) cluster discretely into their own independent 'islands'. The difficulty in smoothly traversing fold space as represented by these databases—e.g., hop from island-to-island or create 'bridges' between islands in fold space—implies that some folds have no well-defined or discernible relationships to others. That is, we miss the weak or more indeterminate (but nevertheless *bona fide*) signals of remote relationships that link distantly-related folds. In addition to the constraints of mutually exclusive clustering, the 3D structural comparisons used in building these databases generally rely upon fairly rigid spatial criteria, such as requiring identical topologies for two entities to group together at the finer (more homologous) classification levels. What relationships might be detectable if we relax the constraints of strict topological identity? As described below, this question is addressed by a recently proposed 'Urfold' model of protein structure [7, 9], which allows for sub–domain-level similarity.

Motivated by striking structure/function similarities across disparate superfamilies, we recently identified relationships between several SFs that exhibit architectural similarity despite topological variability, in a new level of structural granularity that allows for discontinuous fragments and that we termed the 'Urfold' (Fig. 3.1B; [7, 9]). Urfolds[2] were first described in small β-barrel (SBB) domains (Fig. 3.1A) based on

[2]We use the capitalized term 'Urfold' to refer to the concept/theory/model, as a general idea; the lowercase 'urfold' is used when we intend for that specific instance of the word to be limited to

patterns of structure/function similarity (as well as sequence signatures in MSAs, albeit more weakly) in deeply-divergent collections of proteins that adopted either the SH3/Sm or OB superfolds [7]. Notably, the SH3 and OB are two of the most ancient protein folds, and their antiquity is reflected in the fact that they permeate much of information storage and processing pathways—i.e., the transcription and translation apparatus—throughout all domains of life [13, 33].

3.1.3 DeepUrfold: Motivation & Overview

The advent of deep learning, including the application of such approaches to protein sequence and structure representations, affords new opportunities to study protein interrelationships in a wholly different manner—namely, via quantitative comparison of 'latent space' representations of a protein in terms of its lower-dimensional 'embedding'; such embeddings can be at arbitrary levels of granularity (e.g., atomic), and can subsume virtually any types of properties (such as amino acid type, physicochemical features such as electronegatitivty, and phylogenetic conservation of the site). Two powerful features of such approaches are that (i) models can be developed in statistically well-principled manner (or at least strive to be clear about their assumptions), and (ii) models have the capacity to be *integrative*, by virtue of the encoding (or ('featurization') of structural properties alongside phylogenetic, chemical, etc. characteristics of the data (in this case, a protein 3D structure). The present work explores the idea that viewing protein fold space in terms of latent spaces (what regions are populated, with what densities, etc.)—and performing comparative analysis via such spaces (versus in direct or 'real' 3D/geometric space)—is likely to implicitly harbor

a specific case (e.g., "*the* SBB urfold"). Our goal is not to be dogmatic, but rather to be clear and precise as this new concept is being developed.

deep information about protein interrelationships, over a vast multitude of protein evolutionary timescales.

Here, we present a deep learning–based algorithm to systematically identify urfolds, using a new alignment-free, biochemically-aware similarity metric of domain structures based on deep generative models and mixed-membership community detection. We leverage similarities in latent-spaces rather than simple/purely-geometric 3D structures directly, and we can encode any sort of biophysical or other types of properties, thereby allowing more subtle similarities to be detected—such as may correspond to architectural similarities among (dis-)contiguous fragments from different folds or even superfolds (Fig. 3.1C).

3.2 Results

3.2.1 The DeepUrfold Computational Framework: Deep Generative Models

Conventionally, two protein structures that have similar architectures but varying topologies (i.e., folds) are often thought of as resulting from convergent evolution. However, as in the case with the SH3 and OB superfolds, the structure/function similarities [7], and even sequence/structure/function similarities [13], can prove to be quite striking, suggesting that these domain architectures did not arise independently [6, 13] but rather are echoes of a (deep) homology. To study what may be even quite weak 3D similarities, in DeepUrfold we model the evolutionary process giving rise to proteins as an integrated 3D structure/properties 'generator'. In so doing, we seek to learn probability distributions, $p(x|\theta)$, that describe the specific geometries and physicochemical properties of different folds (i.e., features that largely define protein *function*), where the random variable x denotes a single structure drawn from ($x \in \mathbf{x}$) a set of structures labelled as having the same fold ($\mathbf{x}$), and θ denotes the collection of parameters describing the variational distribution over the background (i.e., latent) parameters. We posit that folds with similar probabilistic distributions—which can be loosely construed as ”structure↔function mappings”, under our feature-set—likely have similar geometries/architectures and biophysical properties, regardless of potentially differing topologies (i.e., they comprise an urfold), and that, in turn, may imply a common evolutionary history.

Using the principles of variational inference, DeepUrfold learns the background distribution parameters $\boldsymbol{\theta_i}$ for superfamily distributions, $p_i(x_{ij}|\boldsymbol{\theta_i})$, by constructing and training variational autoencoders (VAE) for each superfamily i and domain structure j (in this work, DeepUrfold is developed using 20 highly-populated SFs from CATH; see Fig 3.1C). The original/underlying posterior distribution, $p_i(x_{ij}|\boldsymbol{\theta_i})$, is unknown and intractable, but it can be approximated by modeling it as an easier-to-learn distribution, $q_i(z_{ij}|\mathbf{x}_i)$; in our case, the approximating distribution $q(z|\mathbf{x})$ is taken as sampling from a Gaussian. To ensure that $q_i(z_{ij}|\mathbf{x}_i)$ optimally describes $p_i(x_{ij}|\boldsymbol{\theta_i})$, we seek to maximize the *evidence lower bound* (ELBO) quantity, which is the lower bound of the marginal likelihood of a single structure, $\ln[p_i(x_{ij})]$. The ELBO inequality can be written as:

$$\begin{aligned} \ln\ p_i(x_{ij}) \geq\ & \mathbb{E}_{q_i(z_{ij}|\mathbf{x}_i)}[\ln\ p_i(x_{ij}|z_{ij})] - \\ & D_{\mathrm{KL}}[q_i(z_{ij}|x_{ij})\,||\,p(z_{ij})] \end{aligned} \tag{3.1}$$

where $p_i(x_{ij})$ is the likelihood, $\mathbb{E}$ is the expectation value of q in terms of p, and $D_{\mathrm{KL}}[q||p]$ is the Kullback-Leibler divergence, or relative entropy, between the two probability distributions q and p. In other words, maximizing the ELBO maximizes the log-likelihood of our learned model, which corresponds to minimizing the entropy or 'distance' (KL divergence) between (i) the true underlying distribution, $p(x|\boldsymbol{\theta})$, and (ii) our learned/inferred posterior distribution of latent parameters given the data, $q(z|\mathbf{x})$. In a similar manner, part of DeepUrfold's testing and development involved training "joint models" using a bag of SFs with *different* topologies, e.g. a mixed SH3 $\cup$ OB set, while accounting for the class imbalance [34, 35] that stems from there being vastly different numbers of available 3D structural data for different protein SFs (e.g., disproportionately abundant immunoglobulin structures).

As input to the VAE, we encode the 3D structure of each protein domain by representing it as a 3D volumetric object, akin to the input used in 3D convolutional neural networks (CNNs); indeed, DeepUrfold's neural network architecture can be viewed as a hybrid 3D CNN-based VAE. In our discretization, atoms are binned into voxels, each of which can be tagged or labeled, atom-wise, with arbitrary properties (biophysical, phylogenetic, etc.). This representation is agnostic of polypeptide chain topology, as the covalent bonding information between residues, and the order of SSEs, is not explicitly retained; note, however, that no information is lost by this representation, as such information is implicit in the proximity of atom-occupied voxels in the model (and can be used to unambiguously reconstruct the 3D structure).

3.2.2 DeepUrfold Models Can Detect Similarities among Topologically -distinct, Architecturally-similar Proteins

To initially assess our SH3, OB and joint SH3/OB DeepUrfold models—and to examine the properties of the Urfold model more broadly—we directly tested the Urfold's

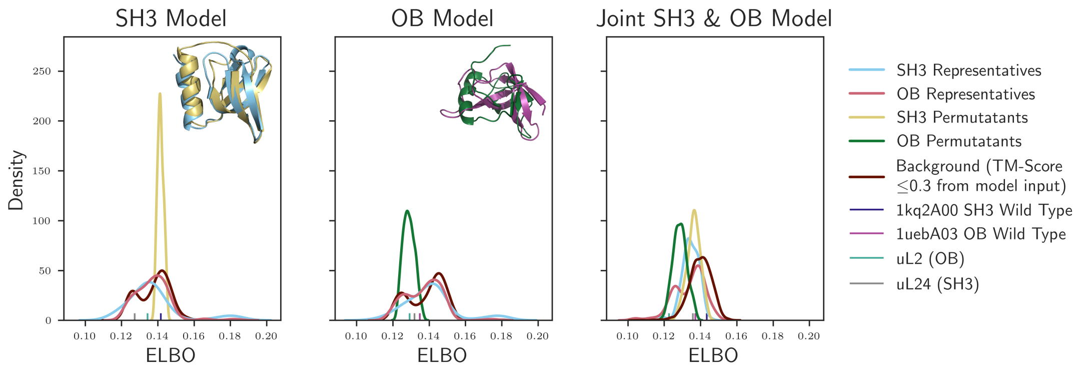


Figure 3.2: Likelihood values can be used to quantify similarities among multi-loop permuted structures. To gauge the sensitivity of our DeepUrfold metric to loop orderings (topology) via generation of fictitious folds, we implemented a multi-loop permutation algorithm ([36]) in order to systematically 'scramble' the SSEs found in an SH3 domain (1k2A00) and an OB domain (1uebA03); in these loop 'rewiring' calculations, we stitched together the SSEs and energetically relaxed the resultant 3D structures using the MODELLER suite. While 96 unique permutations are theoretically possible for a 4-stranded β-sheet [7], only 55 SH3 and 274 OB permuted domains were able to be modeled, presumably because their geometries lie within the radius of convergence of MODELLER (e.g., the loop-creation algorithm did not have to span excessive distances in those cases). Each novel permuted structure was subjected to a DeepUrfold model that had been trained on all other domains from either the **(A)** SH3, **(B)** OB, or **(C)** joint SH3/OB models. Fits to the model were approximated by the ELBO score, which can be viewed as a similarity metric or a measure of 'goodness-of-fit'. In reference to a given model, a given permutant query structure having an ELBO score less than its wild-type structure for that model can be considered as structurally more similar (a better fit) to the model, and thus perhaps more thermodynamically or structurally stable. As reference points, we also include the ELBO scores for ancestrally-reconstructed progenitors of the OB (uL2) and SH3 (uL24) superfolds, based on recent work by [13].

core concept of "*architectural similarity despite topological variability*". This was performed by considering artificial protein domains that have identical architectures but with specifically introduced loop permutations: we obtained these systematic perturbations of a 3D structure's topology by 'rewiring' the SSEs (scrambling the loops), while retaining the overall 3D structure (i.e., architecture). Specifically, (i) we systematically created permuted (fictitious) 3D structures starting with representative

SH3 and representative OB domains (Supp. Fig. 7A) via structural modeling (including energetic relaxation), and (ii) we then subjected these rewired structures, in turn, to each of the SH3, OB and joint SH3/OB DeepUrfold models. Because SBBs typically have six SSEs (five strands and a helix), including four 'core' β-strands, the β-sheet core of an SBB can theoretically adopt one of at least 96 distinct loop permutations [7]; note that, based on the operational definitions/usage of the terms 'topology' and 'fold' in systems such as SCOP, CATH, etc., such engineered permutants almost certainly would be annotated as being from different homologous superfamilies, implying no evolutionary relatedness. Thus, the approach described here is a way to gauge DeepUrfold's ability to discern similarities at the levels of architecture and topology, in a manner agnostic of preexisting classification schemes such as CATH.

In general, we find that the synthetic/permuted domain structures have similar ELBO scores as the corresponding wild-type domains (Fig. 3.2). Those permuted domain structures with ELBO scores *less* than the wild-type domains can be interpreted as being more similar (structurally, biophysically, etc.) to the DeepUrfold variational model, and thus perhaps more thermodynamically stable or structurally robust were they to exist in reality—an interesting possibility as regards protein design and engineering. In terms of more conventional structural similarity metrics, the TM-scores [37] for permuted domain structures against the corresponding wild-type typically lied in the range $\approx 0.3-0.5$—values which would indicate that the permutants and wild-type are not from identical folds, yet are more than just randomly similar (Supp. Fig. 7B).

These findings show that the DeepUrfold model is well-suited to our task because our encoding is agnostic to topological 'connectivity' information and rather is only sensitive to 3D spatial architecture/shape. Even though polypeptide connectivity is implicitly captured in our discretization, our DeepUrfold model intentionally does not consider if two residues are linked by a peptide bond or if two SSEs are contiguous in sequence-space. This approach is useful in finding similarities amongst sets of seemingly dissimilar 3D structures—and thereby identifying specific candidate urfolds—because two sub-domain portions from otherwise rather (structurally) different domains may be quite similar to each other, even if the domains which they are a part of have different (domain-level) topologies but identical overall architectures. This concept can be represented symbolically: for a subset of SSEs, d, drawn from a full domain $\mathcal{D}$, the Urfold model permits relations (denoted by the '$\sim$' symbol) to be detected between two different 'folds', i and j (i.e. $d_i \sim d_j$), without requiring that the relation also be preserved with the stringency of matched topologies at the higher 'level' of the full domain. That is, $d_i \sim d_j \nRightarrow \mathcal{D}_i \sim \mathcal{D}_j$, even though $d_i \subset \mathcal{D}_i$ and $d_j \subset \mathcal{D}_j$ (in contrast to how patterns of protein structural similarity are traditionally conceived, at the domain level). Here, we can view the characteristic stringency or 'threshold' level of the urfold 'd' as being near that of Architecture, while $\mathcal{D}$ reflects both Architecture *and* Topology (corresponding to the classical usage of the term 'fold').

3.2.3 Latent Spaces Capture Gross Structural Properties Across Many Superfamilies, and Reveal a Highly Continuous Nature of Fold Space

The latent space of each superfamily-level DeepUrfold model provides a new view of that SF, and examining the patterns of similarities among such models may offer a uniquely informative view of fold space. Each SF model captures the different 3D geometries and physicochemical properties that characterize that individual SF as a single 'compressed' data point; in this way, the latent space representation (or 'distillation') is more comprehensible than is a full 3D domain structure (or superimpositions thereof). In a sense, the DeepUrfold approach—and its inherent latent space representational model of protein SFs—can reconcile the dichotomy of a continuous versus discrete fold space because the Urfold model (i) begins with no assumptions about the nature of fold space (i.e., patterns of protein interrelationships), and (ii) does not restrictively enforce full topological ordering as a requirement for a relation to be detected between two otherwise seemingly unrelated domains (e.g., $d_i^{\mathsf{SH3}} \sim d_j^{\mathsf{OB}}$ is not forbidden, using the terminology introduced above).

As a first view of fold space through the lens of the Urfold, we use DeepUrfold to compute/represent and analyze the latent spaces of representative domains for highly populated SFs, including mapping the latent spaces into two dimensions (Fig 3.3). Proteins that share similar geometries and biophysical properties will have similar embeddings and should lie close together in this latent-space representation, regardless of the annotated 'true' SF. Though this initial picture of the protein universe is limited to 20 highly populated CATH SFs (in this work), already we can see that these SF domains appear to be grouped and ordered by secondary structure composition (Fig 3.3), consistent with past analyses that used approaches such as multidimensional scaling to probe the overall layout of fold space (e.g., [17]). Compellingly—with respect to the Urfold—variable degrees of intermixing between SFs can be seen in UMAP projections such as illustrated in Fig. 3.3. In addition to this mixing, the latent space projection is not punctate: rather, it is fairly 'compact' (in a loose mathematical sense) and well-connected, with only a few disjoint outlier regions.

Manual inspection of these outlier domain structures shows that many of them are incomplete sub-domains or, intriguingly, a single portion of a larger domain-swapped region [38]. Together, these findings support a rather continuous view of fold space, at least for these 20 exemplary superfamilies.

While each superfamily model is trained independently, with different domain structures (SH3, OB, etc.), we find that the distributions that the VAE-based SF models each learn—again, as 'good' approximations to the true posterior, $p_i(x_{ij}|\boldsymbol{\theta_i})$—are similar, in terms of the dominant features of their latent spaces. In other words, the multiple VAE models (across each unique SF) each learn a structurally low-level, 'coarse-grained' similarity that then yields the extensive overlap seen in Fig. 3.3. When colored by a score that measures secondary structure content, there are clear directions along which the latent-space can be seen to follow, as a gradient from 'all-α' domains to 'all-β' domains, separated by 'α/β' domains. These findings are reassuring with respect to previous studies of protein fold space (e.g., [17]), as well as the geometric intuition that the similarity between two domains would track with their secondary structural content (e.g., two arbitrary all-β proteins are more likely to share geometric similarity than would an all-β and an all-α).

3.2.4 Protein Interrelationships Defy Discrete Clusterings

Our initial finding that protein fold space is rather continuous implies that there are, on average, webs of interconnections (similarities, relationships) between a protein fold $\mathcal{A}$ and its neighbors in fold space ($\mathcal{A}'$, $\mathcal{A}''$, $\mathcal{B}, \ldots$). Therefore, we believe that an optimally realistic view of fold space will not entail hierarchically clustering proteins into mutually exclusive bins. Alternatives to discrete clustering could be such approaches as *fuzzy clustering*, *multi-label classification*, or *mixed-membership community detection* algorithms. DeepUrfold's strategy is to detect communities of similar protein domains, at various levels of stringency, based on the quantifiable similarities of their latent-space representations (versus, e.g., hierarchical clustering based on RMSD or other purely-geometric measures). In DeepUrfold, we formulate this labeling/classification/grouping problem by fitting an edge-weighted [39], mixed-membership [40, 41], hierarchical [42] stochastic block model (SBM; [43]) to a fully connected bipartite graph that is built from the similarity scores between (i) the VAE-based SF-level models (one part of the bipartite graph), and (ii) representative structural domains from the representative SFs (the other part of the bipartite graph). In our case, we weight each edge by the quantity $-\log(\text{ELBO})$ (see Fig 3.1, Eq 3.1). Such a bipartite graph can be represented as an adjacency matrix $\mathbf{A}_{d\times sfam}$ and covariate edge weights $\boldsymbol{x}$ (between vertices in the two 'parts' of the bipartite graph), where $sfam \in 20$ representative SFs and $d \in 3654$ representative domains from 20 representative SFs. The likelihood of such a bipartite graph/network occurring by chance—with the same nodes connected by the same edges with the same weights—is defined by:

$$\begin{split} P(A, x, \gamma, G, k, e, b) = \\ P(A|G)P(x|G,\gamma)P(\gamma|e,b)P(G|k,e,b)P(k|e,b)P(e|b)P(b) \end{split} \tag{3.2}$$

where $\boldsymbol{b}$ is the overlapping partition, $\boldsymbol{e}$ is the matrix of edge counts between groups, $\boldsymbol{k}$ is the labelled degree sequence, and $\boldsymbol{G}$ is a tensor representing half-edges (each edge end-point $r,\ s$) to account for mixed-membership, satisfying $A_{ij} = \sum_{rs} G_{ij}^{rs}$. Edge covariates $\boldsymbol{x}$ are sampled from a microcanonical distribution, $P(x|G,\gamma)$, where γ adds a hard constraint such that $\sum_{ij} G_{ij}^{rs} x_{ij} = \gamma_{rs}$ ([44] Sec. VIIC and personal communication with T. Peixoto).

The parameters for a given SBM are found using Markov chain Monte Carlo (MCMC) methods. Several different models are created for different $\boldsymbol{b}$ and $\boldsymbol{e}$ in order to find the optimal number of blocks with overlapping edges between them, and these are evaluated using a posterior odds-ratio test [40, 41].

Armed with the above SBM methodology, we can now summarize DeepUrfold's overall approach as follows: (i) dataset construction, e.g. via the aforementioned discretization of the 3D structures and biophysical properties into voxelized representations (Draizen et al., in prep); (ii) training of SF-specific models, using VAE-based deep networks; (iii) in an inference stage, calculation of ELBO-based scores for 'fits' obtained by subjecting SF representative i to the VAE model of another SF, $j(\neq i)$; (iv) to detect any patterns amongst these scores, utilization of SBM-based analysis of 'community structure' among the full set of score similarities from the VAE-based SF-level models.

Application of this DeepUrfold methodology to the 20 most highly-populated CATH superfamilies leads us to identify many potential communities of domain structures and SFs (Fig. 3.4). Subjecting all domain representatives to all 20 SF-specific models, in an exhaustive $all_{\text{SF-models}} \times all_{\text{SF-reps}}$ analysis, reveals the overall community structure shown in Fig. 3.4. We argue that two proteins drawn from vastly different SFs (in the sense of their classification in databases such as CATH or SCOP) can share other, more generalized regions of geometric/structural and biophysical properties, beyond simple permutations of secondary structural elements. And, we believe that the minimally-heuristic manner in which the DeepUrfold model is constructed allows it to capture such 'distant' linkages. In particular, these linkages can be identified and quantitatively described as patterns of similarity in the DeepUrfold model's latent space. Clustering domains and superfamilies based on this new similarity metric provides a new view of protein interrelationships—a view that extends beyond simple structural/geometric similarity, towards the realm of integrated structure/function properties.

We find that domains that have similar ELBO scores against various superfamily models (differing from the SF against which they were trained) are more likely to contain important biophysical properties at particular—and, presumably, functionally important—locations in 3D space; these consensus regions/properties can be thought of as 'defining' the domain. Furthermore, if two domains map into the same SBM community, it is likely that both domains share the same scores when run through each SF model (i.e., an inference calculation), so we hypothesize it might contain an urfold that subsumes those two domains (again, agnostic of whatever SFs they are labeled as belonging to in CATH or other databases). We also expect that some domains may be in multiple communities, which may reflect the phenomenon of a protein being constructed of several 'urfold' or sub-domain elements. However, because of the complexities of analyzing, visualizing and otherwise representing such high-dimensional data, in the present work we show only the most likely cluster each domain belongs to.

Given the stochastic nature of the SBM calculation, we ran six different replicates. While each replica produced slightly different hierarchies and numbers of clusters (ranging from 19-23), the communities at the lowest level remained consistent, exhibiting varying degrees of intermixing. In each of the replicates, the SH3 and OB clustered into the same communities as well as Rossman-like and P-loop NTPases, instead of their own individual clusters—consistent with the Urfold view of these particular SFs, as predicted based on manual/visual analysis [9]. In Fig. 3.4, we chose to display the replica with 20 superfamilies and highest overlap score compared to CATH in order to enable easy comparison with CATH. Most notably, each community contains domains from different superfamilies, consistent with the Urfold model (Fig. 3.4A). In the particular subset of proteins treated here, the domains from 'mainly α' and 'α/β' are preferentially associated, while domains from 'mainly β' and 'α/β' group together (Fig. 3.4B) and SH3 and OB cluster together in the same communities (Fig. 3.4A).

In addition to coloring each domain node by CATH superfamily in the circle-packing diagrams, we also explored coloring domain nodes by other types of properties, including (i) secondary structure, (ii) average electrostatic potential, (iii) average partial charge, and (iv) enriched GO terms (Supp. Fig. 12-17); a navigable, web-based interface for exploring these initial DeepUrfold results is freely available at https://bournelab.org/research/DeepUrfold/. Interestingly, domains with similar average electrostatic potentials (Supp. Fig. 12) and partial charges (Supp. Fig. 13) can be found to cluster into similar groups, whereas the CATH-based circle-packing diagrams, colored by those same features, have no discernable order or structuring; whether or not this phenomenon stems from any underlying, functionally-relevant 'signal' is a question of interest in further work.

In order to assess how 'well' our DeepUrfold model does, we compare our clustering results to CATH. However, we emphasize that there is no reliable, objective ground truth for a map of fold space, as there is no universally-accepted, 'correct' description of fold space (and, it can be argued, even 'fold'). Therefore, we compare our DeepUrfold results to a well-established system (e.g., CATH) with the awareness that these are two fundamentally different approaches to representing and describing the protein universe. Indeed, because our model uses a different input representation of proteins that intentionally ignores all topological/connectivity information, we expect that our model will be least similar to CATH in terms of SBM-related measures such as partition overlap, homogeneity, and completeness [41]. Given all this, models that differ from CATH—versus matching or recapitulating it—can be considered as representing an alternative view of the protein universe. Somewhat counterintuitively, we deem poorer comparison metrics (e.g., less similarity to CATH) as providing stronger support for the Urfold model of protein structure. Simultaneously, we compare how well other, independently-developed sequence- and structure-based models can reconstruct CATH (Fig. 3.5). Among all these methods, our DeepUrfold approach produces results are the most divergent from CATH, consistent with DeepUrfold's approach of taking a wholly new view of the protein universe and the domain-level structural similarities that shape it. We also show that many other algorithms have difficulty reconstructing CATH, possibly due to the extensive manual curation of CATH, but much more closely reproduce CATH than does our method—we suspect that this is due, in large part, to DeepUrfold's incorporation and integration of more *types* of information than purely 3D geometry.

3.3 Discussion, Further Outlook

This work has presented a new deep learning-based approach, termed 'DeepUrfold', aimed at systematically identifying putative new urfolds. Notably, the DeepUrfold framework (i) is sensitive to 3D structure and structural similarity between pairs of proteins, but is minimally heuristic (e.g., it does not rely upon pre-set RMSD thresholds or the like) and, most notably, is alignment-free (as it leverages latent-space embeddings of structure, versus direct 3D coordinates, for comparison purposes); (ii) beyond the residue-level geometric information defining a 3D structure (i.e. coordinates), DeepUrfold is an extensible model insofar as it can incorporate *any* types of properties of interest (so long as they can be encoded in a deep model), e.g. biophysical and physicochemical characteristics (electrostatic charge, solvent exposure, etc.), site-by-site phylogenetic conservation, and so on; (iii) the method provides a quantitative metric, in the form of the deep neural network's loss function (at the inference stage), that is amenable to approaches that are more generalized than brute-force hierarchical clustering (e.g., using loss function scores in stochastic block modeling to construct mixed-membership communities of proteins). In the above ways, DeepUrfold can be viewed as an integrative approach that, while motivated by structural (dis)similarities across fold space, is also cognizant of *sequence↔structure↔function* interrelationships. This is intentional: molecular evolution acts on the sequence/structure/function triad as its base 'entity', not on purely geometric/3D structure alone. We believe that any purely geometric/structure-based approach will be similarly constrained in its ability to accurately represent fold space.

We demonstrate (i) the general utility of this new type of similarity metric for representing and comparing protein domain structures, based on deep generative models, and (ii) that a mixed-membership community detection algorithm can identify what we previously found, via manual/visual analysis [9], to be putative urfolds. Finally, we emphasize that because DeepUrfold is agnostic of precise protein topology (i.e., order of SSEs in 3-space), higher levels of similarity can be readily detected (above CATH's 'T' level, below its 'A' level), including the potential of non-contiguous fragments. We believe that such such spatially-compact groups of frequently recurring sub-domain fragments, sharing similar architectures (independent of topology) within a given group—which, again, we term an 'urfold'—could correspond to primitive 'design elements' in the early evolution of protein domains [19]. We note that [45] has made similar points.

Overall, the DeepUrfold framework provides a sensitive approach to detect and thus explore distant protein inter-relationships, which we suspect correspond to weak phylogenetic signals (perhaps as echoes of remote/deep homology). Also notable, the embeddings produced by our VAE models and ELBO similarity scores provide new methods to visualize and interpret protein interrelationships on the scale of a full fold space. From these models, it is clear that there is a fair degree of continuity between proteins in fold space, and intermixing between what has previously been labeled as separate superfamilies; a corollary of this finding is that discretely clustering protein embeddings is ill-advised from this perspective of a densely-populated, smoother-than-expected fold space. An open question is the degree to which the extent of overlap between individual proteins (or groups of proteins, as an urfold) in this fold space is reflective of underlying evolutionary processes, e.g. akin to [18]'s finding that "evolutionary information is encoded along these structural bridges [in fold space]".

An informative next step would be to use DeepUrfold to identify structural fragments that contain similar patterns of geometry and biophysical properties between proteins from very different superfamilies. Notably, these fragments may be continuous or discontinuous, and pursuing this goal might help unify the 'primordial peptides' [2] and 'themes' [24] concepts with the Urfold hypothesis, allowing connections between unexplored (or at least under-explored) regions of fold space. We suspect that 'Explainable AI' techniques, such as Layer-wise Relevance Propagation (LRP; [46, 47]), can be used to elucidate which atoms/residues, along with their 3D locations and biophysical properties, are deemed most important in defining the various classification groups (i.e., into urfold $\mathcal{A}$ versus urfold $\mathcal{B}$). This goal can be pursued within the DeepUrfold framework because we discretize full domain structures into voxels: thus, we can probe the neural network to learn about specific voxels, or groups of specific voxels (e.g., amino acid residues), that contribute as sub-domain structural elements. Doing so would, in turn, be useful in finding common sub-domain segments from different superfamilies. We hypothesize that the most 'relevant' (in the sense of LRP) voxels would highlight important sub-structures; most promisingly, that we know the position, physicochemical and biophysical properties, and so on about the residues would greatly illuminate the *physical* basis for the deep learning-based classification. In addition, this would enable us to explore in more detail the mechanistic/structural basis for the mixed-membership features of the SBM-based protein communities. Such communities—beyond helping to detect and define new urfolds—may offer a novel perspective on remote protein homology.

3.4 Computational Methodology

3.4.1 Datasets

Using Prop3D, a computational toolkit that we have been developing for handling protein properties in machine learning and structural bioinformatics pipelines (Draizen et al., in prep), we have now created a 'Prop3D-20sf' dataset. This dataset uses 20 CATH superfamilies of interest (Fig. 3.1C; Supp. Table 1). Domain structures from each of the 20 SFs are 'cleaned' by adding missing residues with MODELLER [48], missing atoms with SCWRL4 [49], and protonating and energy minimizing (simple de-bump) with PDB2PQR [50]. Next, we compute a host of derived properties for each domain in CATH (Draizen et al., in prep)–including (i) purely geometric/structural quantities, e.g. secondary structure [51], solvent accessibility, (ii) physicochemical properties, e.g. hydrophobicity, partial charges, electrostatic potentials, and (iii) basic chemical descriptors (atom and residue types). The computation was performed using the Toil workflow engine [52] and data was stored using the Hierarchical Data Format (version 5) in the Highly Scalable Data Service (HSDS). The domains from each superfamily were split such that all members of a S35 35% sequence identity cluster (pre-calculated by CATH) were on the same side of the split. We split them roughly 80% training, 10% validation, and 10% test (Draizen et al., in prep; https://doi.org/10.5281/zenodo.6873024).

In Prop3D-20sf, each atom is attributed with the following seven groups of features that are one-hot (Boolean) encoded: (1) Atom Type (C,CA,N,O,OH,Unknown); (2) Residue Type (ALA, CYS, ASP, GLU, PHE, GLY, HIS, ILE, LYS, LEU, MET, ASN, PRO, GLN, ARG, SER, THR, VAL, TRP, TYR, Unknown); (3) Secondary Structure (Helix, Sheet, Loop/Unknown); (4) Hydrophobic (or not); (5) Electronegative (or not); (6) Positively Charged (or not); and (7) Solvent-exposed (or not). However, for all final production models reported here, the ”residue type” feature was omitted as it was found to be uninformative, at least for this type of representation (Supp. Fig. 3).

3.4.2 Protein Structure Representation

We represent protein domains as voxels, or 3D volumetric pixels. Briefly, our method centers protein domains in a 256^3 ^{3} cubic volume to allow for large domains, and each atom is mapped to 1 ^{3} voxels using a *k*D-tree data structure, with a query ball radius set to the van der Waals radius of the atom. If two atoms share the space in a given voxel, the maximum between their feature vectors is used (justifiable because they are all binary-valued). Because a significant fraction of voxels in our representation domain do not contain any atoms, protein domain structures can be encoded via a sparse representation; this substantially mitigates the computational costs of our deep learning workflow using MinkowskiEngine [53].

Because there is no unique or 'correct' orientation of a protein structure, we applied random rotations to each protein domain structure; these rotations were in the form of orthogonal transformation matrices randomly drawn from the Haar distribution, which is the uniform distribution on the 3D rotation group (i.e., SO(3); [54]).

3.4.3 VAE Model Design and Training

A sparse 3D-CNN variational autoencoder was adapted from MinkowskiEngine ([53, 55]). In the Encoder, there are 7 blocks consisting of Convolution (n->2n), BatchNorm, ELU, Convolution (2n->2n), BatchNorm, and ELU, where n=[16,32,64,128,256,512,1024], doubling at each block. Finally, the tensors are pooled using Global pooling, and the model outputs both a normal distribution's mean and log variance. Next, the learned distribution is sampled from and used as input into the Decoder. In the decoder, there are also 7 blocks, where each block consists of ConvolutionTranspose(2n->n), BatchNorm, ELU, Convolution(n->n), BatchNorm, and ELU. Finally, one more convolution is used to output a reconstructed domain structure in a 264^{33} volume.

In VAEs, a 'reparameterization trick' allows for backpropagation through random variables by making only the mean (μ) and variance (σ) differentiable, with a random variable that is normally distributed ($\mathcal{N}(0, \mathbf{I})$). That is, the latent variable posterior $\mathbf{z}$ is given by $\mathbf{z} = \mu + \sigma \bigodot \mathcal{N}(0, \mathbf{I})$, where $\bigodot$ denotes the Hadamard (element-wise) matrix product and $\mathcal{N}$ is the 'auxiliary noise' term ([56]).

We optimize against the Evidence Lower BOund (ELBO) described in equation 3.1, which combines (i) the mean squared error (MSE) of the reconstructed domain and (ii) the difference between the learned distribution and the true distribution of the SF (i.e., the KL-divergence, or relative entropy; [56]).

We used stochastic gradient descent (SGD) as the optimization algorithm, with a momentum of 0.9 and 0.0001 weight decay. We began with a learning rate of 0.2 and decreased its value by 0.9 every epoch using an exponential learning rate scheduler. Our final network has $\approx$110M parameters in total and all the networks were trained for 30 epochs, using a batch size of 255. We utilized the open-source frameworks PyTorch [57] and PytorchLightning [58] to simplify training and inference, and to make the models more reproducible.

In order to optimize hyperparameters for the VAE, we used Weights & Biases Sweeps [59] to scan over the batch size, learning rate, convolution kernel size, transpose convolution kernel size, and convolution stride in the Ig model, while optimizing the ELBO. We used the Bayesian Optimization search strategy and hyperband method with 3 iterations for early termination. We found no significant changes and used the default values: convolution kernel size of 3, transpose convolution kernel size of 2, and convolution stride of 2.

Due to a large-scale class imbalance between the number of domains in each superfamily, we follow the "one-class classifier" approach, creating one VAE for each superfamily. We also train a joint SH3 and OB model and compare random over- and under-sampling from ImbalancedLearn [35] on joint models of multiple superfamilies (Supp. Fig. 8).

All 20 models used throughout this work were trained using 1-4 NVIDIA RTX A6000 GPUs.

3.4.4 Evaluation of Model Performance

We calculate the area under the receiver operating characteristic curve (auROC) and the area under the precision-recall curve (auPRC) for 20 SFs. Representative domains, as defined by CATH, for each superfamily were subjected to their SF-specific VAE and predicted values were micro-averaged to perform auROC and auPRC calculations. Immunoglobulins were chosen to display in the supplemental material for this paper (Supp. Fig. 4-6), but the results for all SFs can be found in the extended supplemental material. All SFs report similar metrics for each group of features.

3.4.5 Assess the Urfold Model by Subjecting Proteins with Permuted Secondary Structures to Superfamily-specific VAEs

To gauge the sensitivity of our DeepUrfold model to loop orderings (i.e., topology), we generate fictitious folds by implementing a multi-loop permutation algorithm [36] in order to 'scramble' the secondary structural elements (SSEs) found in a representative SH3 and OB domains. We stitch together the SSEs and relax the conformations/energetics of each new 3D structure using the MODELLER suite [48].

Next, each novel permuted structure is subjected to a VAE model trained on all other domains from the SH3 homologous superfamily. Fit to the model is approximated by the log likelihood score of the permuted and natural (wild-type) protein represented

ELBO scores, which can be viewed as a similarity metric. We also calculate a ‘background’ distribution of each model by perming an all vs all TM-align for all domains in our representative CATH domains, saving domain that have a TM-Score ≤ 0.3 as that is thought to represent domains that have random similarity.

3.4.6 Latent-space Organization

We subject representative domains from a single superfamily through its superfamily model and visualize the latent space of each representative. A ‘latent-space’ for a given domain corresponds to a 1024 dimensional vector describing the representatives in their most ‘compressed’ form, accounting for the position of each atom and their biophysical properties represented by the mean of the learned distribution. We combine the latent spaces from each domain from each superfamily and then reduce the number of dimensions to two in order to easily visualize it; the latter is achieved using the uniform manifold approximation and projection (UMAP) algorithm. UMAP is a dimensionality reduction algorithm that is similar to methods such as PCA (principal component analysis; Supp. Fig. 9) and particularly t-SNE (t-distributed stochastic neighbor embedding; Supp. Fig. 10), with the benefit of preserving topological relationships at both local and global scales in a dataset.

3.4.7 Mixed-membership Community Detection

We performed all-vs-all comparisons of domains and superfamilies by subjecting representative protein domain structures from each of the 20 chosen SF through each SF-specific one-class VAE model. The ELBO loss score for each domain—SF-model pair can be used to quantitatively evaluate pairwise ’distances’ between SFs by treating it as a fully connected bipartite graph between domains and SF models, defined by adjacency matrix $\boldsymbol{A}_{ij}$, with edges weighted by the -log(ELBO) score in covauate matrix $\boldsymbol{x}$. Stochastic Block Models (SBM; [43]) are a generative model for random graphs that can be used to partition the bipartite graph into communities of domains that have similar distribution of edge covariates between them [39]. Using the SBM liklihood equation (equation 3.2), inference is done via the posterior:

$$P(b, G|A, x) = \frac{P(A, x, , G, k, e, b)}{P(A, x)} \tag{3.3}$$

where $\boldsymbol{b}$ is the overlapping partition, $\boldsymbol{e}$ is the matrix of edge counts between groups, $\boldsymbol{k}$ is the labelled degree sequence, and $\boldsymbol{G}$ is a tensor representing half-edges (each edge end-point r, s) to account for mixed-membership, satisfying $A_{ij} = \sum_{rs} G_{ij}^{rs}$. Edge covariates $\boldsymbol{x}$ are sampled from a microcanonical distribution, $P(x|G, \gamma)$, where γ adds a hard constraint such that $\sum_{ij} G_{ij}^{rs} x_{ij} = \gamma_{rs}$ ([44] Sec. VIIC and personal communication with T. Peixoto).

Using the same SBM approach as we did for DeepUrfold, we compare our results to state-of-the-art sequence- and structure-based methods for comparing proteins. All SBMs are created using fully connected bipartite graphs connected n CATH S35 domains to m Superfamily models. In this case, we used 3654 representative CATH domains from 20 superfamilies, creating a 3654×20 similarity matrix for each method we wish to compare. Each SBM was degree corrected, overlapping, and nested and fit to a real normal distribution of edge covariates. For methods with decreasing scores (closer to zero is best), we took the negative log of each score, whereas scores from methods with increasing scores remained the same.

While only the ‘Superfamily-specific‘ models are directly comparable (e.g. where $n \times m$ matrices are the original output created by subjecting n CATH representative domains without labels to m superfamily-specific models), we also included ‘Pairwise’ and ‘Single Model’ methods. For pairwise approaches, an all-vs-all $n \times n$ similarity matrix is created and is converted to an $n \times m$ by taking the median distance of a single CATH domain to every other domain in a given superfamily. ‘Single Model’ approaches are where a single model is trained on all known proteins and outputs a single embedding score for each domain, creating an $n \times 1$ vector. To convert it into an $n \times m$ matrix, we take the median distance of a single CATH domain embedding to every other domain embedding from a given superfamily.

3.4.8 Comparisons to CATH

Because we have no ground truth with the Urfold view of the protein universe, we perform cluster comparison metrics on each SBM community compared to the original CATH clusterings; these measures can include partition overlap, homogeneity, and completeness for each of the protein comparison tools:

- **Silhouette Score:** measure of how similar an object is to its own cluster (cohesion) compared to next closest cluster (separation). -1: incorrect, 0: perfect, 1: too dense
- **Overlap:** maximum overlap between partitions by solving an instance of the maximum weighted bipartite matching problem [41]
- **Homogeneity:** each cluster contains only members of a single class. [0, 1], 1=best
- **Completeness:** all members of a given class are assigned to the same cluster. [0, 1], 1=best

All comparisons start using the sequence and structure representatives from CATH's S35 cluster for each of the 20 superfamilies of interest. USEARCH [60] was run twice with parameters `-allpairs_local` and `-allpairs_global`; both runs included the `-acceptall` parameter. HMMER [61] models were built using (1) MUSCLE [62] alignments from CATH's S35 cluster; and (2) a deep MSA created from EVcouplings [63] using jackhmmer [61] and UniRef90 of the first S35 representative for each superfamily. Each HMMER model was used to search all representatives, reporting all sequences with bitscores $\geq -10^{12}$. SeqDesign [64] was run using the same MSAs from EVcouplings. We also compared against the pretrained ESM models [65].

For other structure-based comparisons, we ran TM-Align [66] on all representative domains with and without circular permutations saving RMSD and TM-Scores. Struct2Seq [67] was run with default parameters after converting domain structure representatives into dictionaries matching the required input.

3.5 Data Availability

The Prop3D dataset used to train each superfamily model can found at `https://doi.org/10.5281/zenodo.6873024`, which includes the raw HDF file as well as instructions to access the public version of the dataset on the University of Virginia Research Computing HSDS endpoint `http://hsds.uvarc.io` (Draizen et al., in prep).

The extended supplemental material, including the 20 pre-trained SF models and raw output from the stochastic block modelling of DeepUrfold and other tools used to compare against can be found at `https://doi.org/10.5281/zenodo.6916524`.

All code to build datasets and train models can be found at `http://github.com/bouralab/Prop3D` and `http://github.com/bouralab/DeepUrfold`, respectively.

We also provide an accompanying website to explore the SBM communities and the CATH hierarchy at `https://bournelab.org/research/DeepUrfold/`

Acknowledgements

We thank Luis Felipe Murillo and John Readey for support with HSDS, as well as Jaime Iranzo and Tiago Peixoto for discussions regarding stochastic block models; we also thank Menuka Jaiswal, Saad Saleem, and Yonghyeon Kweon for early efforts on this project. Portions of this work were supported by the University of Virginia and NSF Career award MCB-1350957. EJD was supported by a University of Virginia Presidential Fellowship in Data Science.

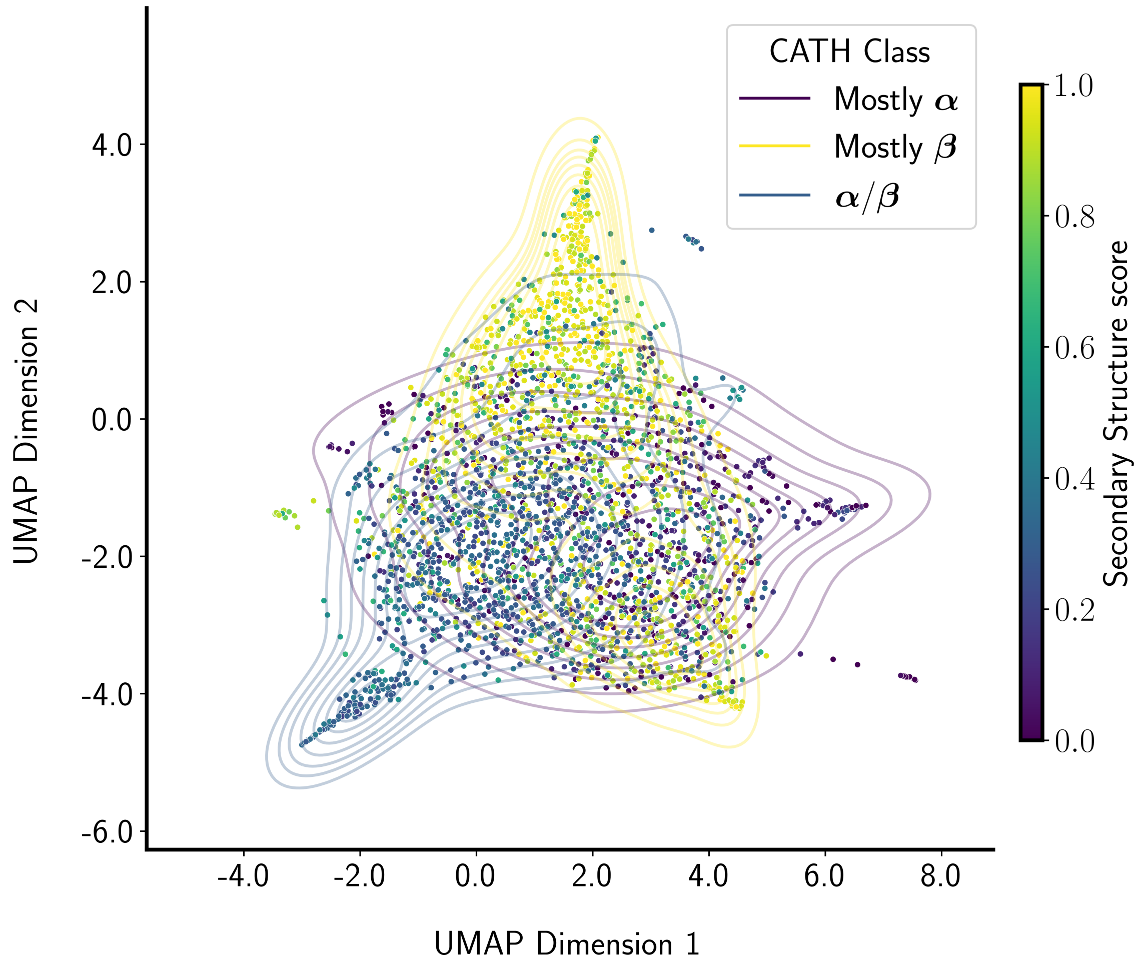


Figure 3.3: Dominant variables of DeepUrfold's latent-space models capture gross structural properties and indicate a highly continuous fold space. In a pilot study, we used DeepUrfold to develop 20 distributions/models for 20 CATH homologous superfamilies. Representatives from each SF were subjected to deep models that were trained on domains from the same SF, and then the latent space variables for each structural domain were examined via the uniform manifold approximation and projection (UMAP) method, thereby reducing the 1024 dimensions of the actual model to the two-dimensional projection shown here. In this representation, kernel density estimates (isodensity contour lines) surround domains with the same annotated CATH *Class*. Each domain is colored by its secondary structure score, computed as $\frac{\#\beta \text{ atoms } - \#\alpha \text{ atoms}}{2(\#\beta \text{ atoms } + \#\alpha \text{ atoms})} + 0.5$. The protein domains can be seen to group together by secondary structure composition; moreover, they are roughly ordered, with the α/β region extensively overlapping the mostly-β region (yellow, predominantly in the vertical direction) and mostly-α region (purple, running predominantly horizontally).

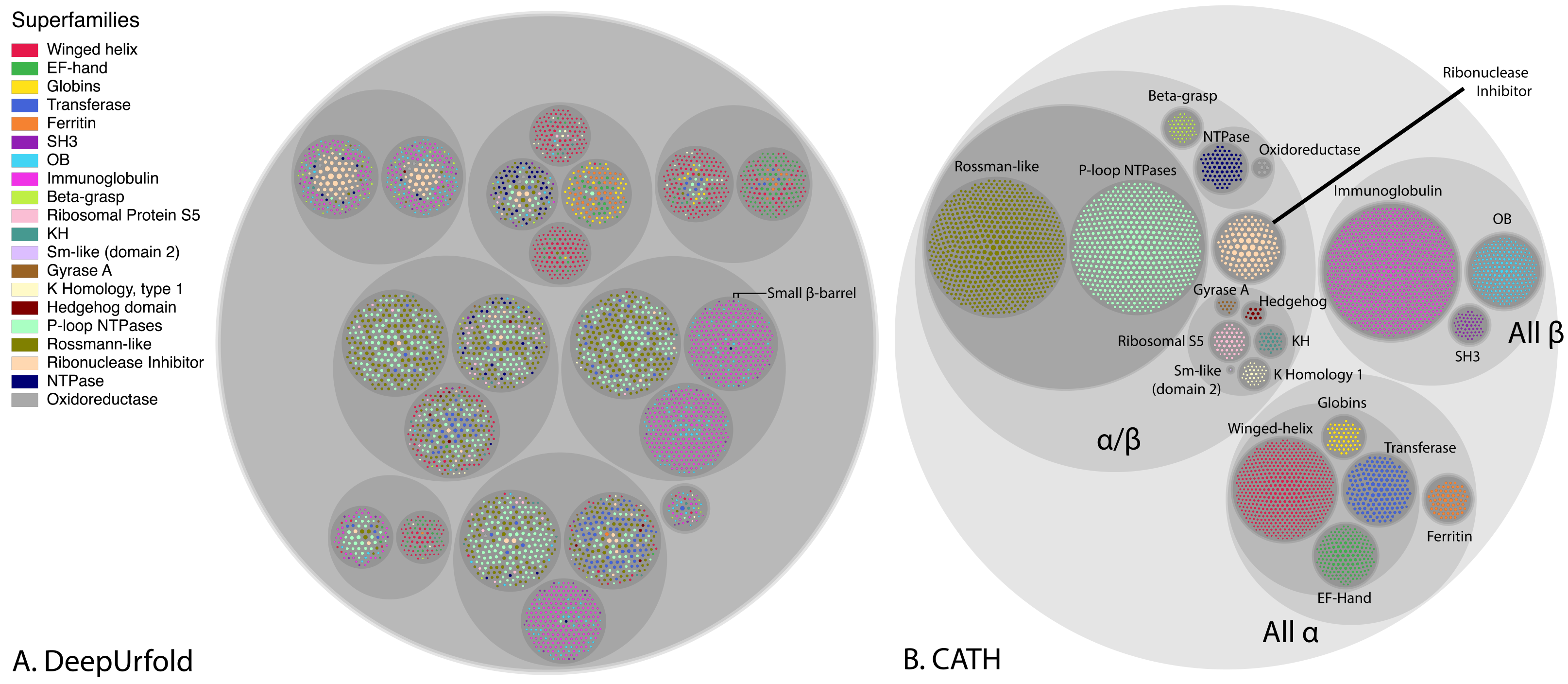


Figure 3.4: Protein interrelationships defy discrete clusterings: Stochastic block modeling of an all-vs-all comparison of domain structures and superfamily models. A) We represent the the SBM communities predicted by DeepUrfold as a circle packing diagram following the same hierarchy. Each domain is displayed as the inner most circles (leafs) colored by the annotated CATH superfamily and sized by their number of atoms. All of the superfamily labelled nodes clustered together and were removed from this list (See supplemental file 2). As proof of concept, we show the SH3 and OB domains are found within the same communities. B) CATH Hierarchy represented as a circle packing diagram showing that DeepUrfold is learning a completely different hierarchy.

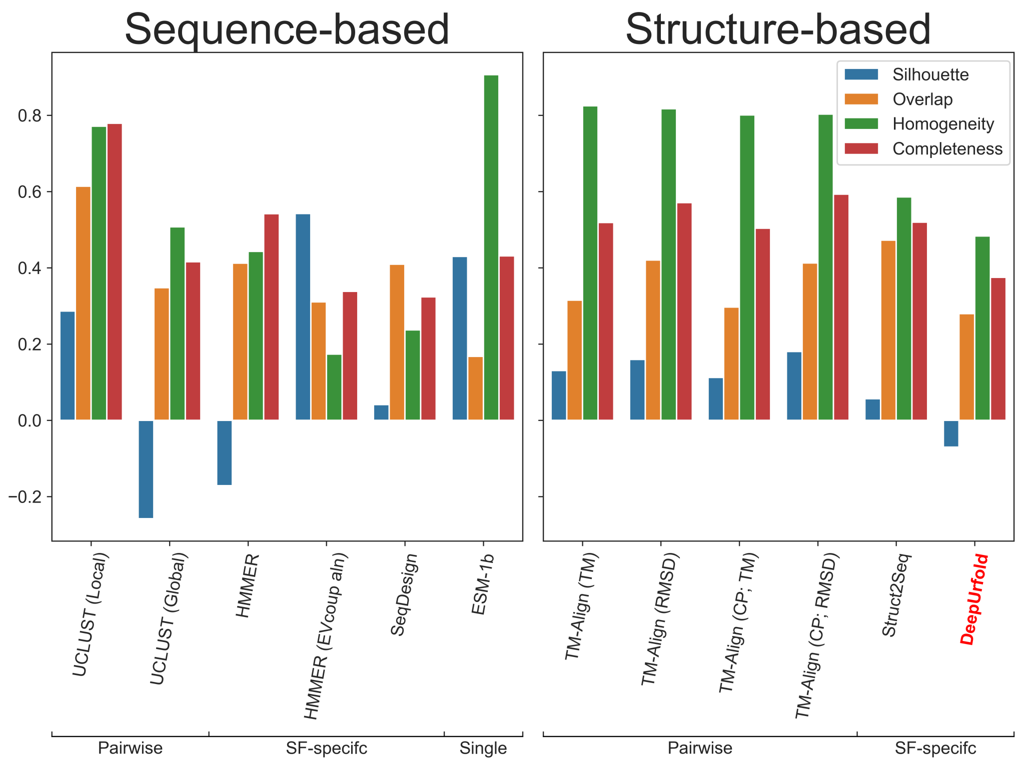


Figure 3.5: DeepUrfold does not recapitulate CATH. We compare DeepUrfold to other sequence- and structure-based protein similarity tools by attempting to reconstruct CATH. The scores from each of the algorithms are used as edge weights in the SBM. If scores were increasing e.g. were a distance metric, the converted to a similarity metric by -x or -log(x). We take the communities at the lowest hierarchical level as clusters and use cluster comparison metrics to understand how well each algorithm/similarity metric can be used to recapitulate CATH. For each metric of Silhouette Score, overlap, homogeneity, and completeness, a value of 1 is deemed best. DeepUrfold does poorly based for each metric because it does not produce the same clusters, and is learning something completely different compared to the other algorithms. For TM-Align, 'CP' stands for Circular Permutation. For more information, see Supp Table 2.

Bibliography

[1] Rachel Kolodny, Leonid Pereyaslavets, Abraham O Samson, and Michael Levitt. On the universe of protein folds. *Annual Review of Biophysics*, 42:559–582, Mar 2013.

[2] Vikram Alva, Johannes Söding, and Andrei N Lupas. A vocabulary of ancient peptides at the origin of folded proteins. *eLife*, 4:e09410, dec 2015.

[3] Rachel Kolodny, Sergey Nepomnyachiy, Dan S Tawfik, and Nir Ben-Tal. Bridging themes: short protein segments found in different architectures. *Molecular Biology and Evolution*, 38(6):2191–2208, may 2021.

[4] Yana Bromberg, Ariel A. Aptekmann, Yannick Mahlich, Linda Cook, Stefan Senn, Maximillian Miller, Vikas Nanda, Diego U. Ferreiro, and Paul G. Falkowski. Quantifying structural relationships of metal-binding sites suggests origins of biological electron transfer. *Sci. Adv.*, 8(2), jan 2022.

[5] Claudia Alvarez-Carreño, Rohan J Gupta, Anton S. Petrov, and Loren Dean Williams. The evolution of protein folds by creative destruction. *bioRxiv*, 2022.

[6] Philippe Youkharibache. Protodomains: Symmetry-related supersecondary structures in proteins and self-complementarity. *Methods in Molecular Biology*, 1958:187–219, 2019.

[7] Philippe Youkharibache, Stella Veretnik, Qingliang Li, Kimberly A Stanek, Cameron Mura, and Philip E Bourne. The small β-barrel domain: A survey-based structural analysis. *Structure*, 27(1):6–26, jan 2019.

[8] Cameron Mura, Peter S. Randolph, Jennifer Patterson, and Aaron E. Cozen. Archaeal and eukaryotic homologs of Hfq: A structural and evolutionary perspective on Sm function. *RNA Biology*, 10(4):636–651, 2013.

[9] Cameron Mura, Stella Veretnik, and Philip E Bourne. The Urfold: Structural similarity just above the superfold level? *Protein Science*, 28(12):2119–2126, nov 2019.

[10] N V Grishin. Fold change in evolution of protein structures. *Journal of Structural Biology*, 134(2-3):167–185, jun 2001.

[11] Lisa N Kinch and Nick V Grishin. Evolution of protein structures and functions. *Current Opinion in Structural Biology*, 12(3):400–408, jun 2002.

[12] S Sri Krishna and Nick V Grishin. Structural drift: a possible path to protein fold change. *Bioinformatics*, 21(8):1308–1310, apr 2005.

[13] Claudia Alvarez-Carreño, Petar I Penev, Anton S Petrov, and Loren Dean Williams. Fold evolution before LUCA: Common ancestry of SH3 domains and OB domains. *Molecular Biology and Evolution*, 38(11):5134–5143, oct 2021.

[14] Ruslan I Sadreyev, Bong-Hyun Kim, and Nick V Grishin. Discrete-continuous duality of protein structure space. *Current Opinion in Structural Biology*, 19(3):321–328, jun 2009.

[15] William R Taylor. Exploring protein fold space. *Biomolecules*, 10(2), jan 2020.

[16] L Holm and C Sander. Mapping the protein universe. *Science*, 273(5275):595–603, aug 1996.

[17] Jingtong Hou, Se-Ran Jun, Chao Zhang, and Sung-Hou Kim. Global mapping of the protein structure space and application in structure-based inference of

protein function. *Proceedings of the National Academy of Sciences of the United States of America*, 102(10):3651–3656, mar 2005.

[18] Hannah Edwards and Charlotte M Deane. Structural bridges through fold space. *PLoS Computational Biology*, 11(9):e1004466, sep 2015.

[19] Jeffrey Skolnick, Adrian K Arakaki, Seung Yup Lee, and Michal Brylinski. The continuity of protein structure space is an intrinsic property of proteins. *Proceedings of the National Academy of Sciences of the United States of America*, 106(37):15690–15695, sep 2009.

[20] Iddo Friedberg and Adam Godzik. Fragnostic: walking through protein structure space. *Nucleic Acids Research*, 33(Web Server issue):W249–51, jul 2005.

[21] Andrew Harrison, Frances Pearl, Richard Mott, Janet Thornton, and Christine Orengo. Quantifying the similarities within fold space. *Journal of Molecular Biology*, 323(5):909–926, nov 2002.

[22] Andrew Harrison, Frances Pearl, Ian Sillitoe, Tim Slidel, Richard Mott, Janet Thornton, and Christine Orengo. Recognizing the fold of a protein structure. *Bioinformatics*, 19(14):1748–1759, sep 2003.

[23] Alexander Goncearenco, Alexey K Shaytan, Benjamin A Shoemaker, and Anna R Panchenko. Structural perspectives on the evolutionary expansion of unique protein-protein binding sites. *Biophysical Journal*, 109(6):1295–1306, sep 2015.

[24] Sergey Nepomnyachiy, Nir Ben-Tal, and Rachel Kolodny. Complex evolutionary footprints revealed in an analysis of reused protein segments of diverse lengths. *Proceedings of the National Academy of Sciences of the United States of America*, 114(44):11703–11708, oct 2017.

[25] Martin Steinegger, Markus Meier, Milot Mirdita, Harald Vöhringer, Stephan J Haunsberger, and Johannes Söding. HH-suite3 for fast remote homology detection and deep protein annotation. *BMC Bioinformatics*, 20(1):473, sep 2019.

[26] Inbal Budowski-Tal, Yuval Nov, and Rachel Kolodny. FragBag, an accurate representation of protein structure, retrieves structural neighbors from the entire PDB quickly and accurately. *Proceedings of the National Academy of Sciences of the United States of America*, 107(8):3481–3486, feb 2010.

[27] Rachel Kolodny, Patrice Koehl, Leonidas Guibas, and Michael Levitt. Small libraries of protein fragments model native protein structures accurately. *Journal of Molecular Biology*, 323(2):297–307, oct 2002.

[28] Janani Durairaj, Mehmet Akdel, Dick de Ridder, and Aalt D J van Dijk. Geometricus represents protein structures as shape-mers derived from moment invariants. *Bioinformatics*, 36(Suppl_2):i718–i725, dec 2020.

[29] Ian Sillitoe, Natalie Dawson, Tony E Lewis, Sayoni Das, Jonathan G Lees, Paul Ashford, Adeyelu Tolulope, Harry M Scholes, Ilya Senatorov, Andra Bujan, Fatima Ceballos Rodriguez-Conde, Benjamin Dowling, Janet Thornton, and Christine A Orengo. CATH: expanding the horizons of structure-based functional annotations for genome sequences. *Nucleic Acids Research*, 47(D1):D280–D284, jan 2019.

[30] Antonina Andreeva, Dave Howorth, Cyrus Chothia, Eugene Kulesha, and Alexey G Murzin. SCOP2 prototype: a new approach to protein structure mining. *Nucleic Acids Research*, 42(Database issue):D310–4, jan 2014.

[31] Naomi K Fox, Steven E Brenner, and John-Marc Chandonia. SCOPe: Structural classification of proteins–extended, integrating SCOP and ASTRAL data

and classification of new structures. *Nucleic Acids Research*, 42(Database issue):D304–9, jan 2014.

[32] Hua Cheng, R Dustin Schaeffer, Yuxing Liao, Lisa N Kinch, Jimin Pei, Shuoyong Shi, Bong-Hyun Kim, and Nick V Grishin. ECOD: an evolutionary classification of protein domains. *PLoS Computational Biology*, 10(12):e1003926, dec 2014.

[33] Vishal Agrawal and Radha KV Kishan. Functional evolution of two subtly different (similar) folds. *BMC Structural Biology*, 1(1):1–6, 2001.

[34] Ronaldo C. Prati, Gustavo E. A. P. A. Batista, and Maria Carolina Monard. Data mining with imbalanced class distributions: Concepts and methods. 2009.

[35] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[36] Liang Dai and Yaoqi Zhou. Characterizing the existing and potential structural space of proteins by large-scale multiple loop permutations. *Journal of Molecular Biology*, 408(3):585–595, may 2011.

[37] Jinrui Xu and Yang Zhang. How significant is a protein structure similarity with tm-score = 0.5? *Bioinformatics*, 26(7):889–895, Apr 2010.

[38] Yanshun Liu and David Eisenberg. 3D Domain swapping: As domains continue to swap. *Protein science*, 11(6):1285–1299, 2002.

[39] Tiago P Peixoto. Nonparametric weighted stochastic block models. *Physical review. E*, 97(1-1):012306, jan 2018.

[40] Tiago P. Peixoto. Model selection and hypothesis testing for large-scale network models with overlapping groups. *Physical Review X*, 5(1), mar 2015.

[41] Tiago P. Peixoto. Revealing consensus and dissensus between network partitions. *Physical Review X*, 11(2):021003, apr 2021.

[42] Tiago P. Peixoto. Hierarchical block structures and high-resolution model selection in large networks. *Physical Review X*, 4(1), mar 2014.

[43] Tiago P. Peixoto. Nonparametric Bayesian inference of the microcanonical stochastic block model. *Physical Review E*, 95(1), jan 2017.

[44] Tiago P. Peixoto. Bayesian stochastic blockmodeling. pages 289–332, nov 2019.

[45] Rachel Kolodny. Searching protein space for ancient sub-domain segments. *Current Opinion in Structural Biology*, 68:105–112, 2021.

[46] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: An overview. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors, *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700 of *Lecture notes in computer science*, pages 193–209. Springer International Publishing, Cham, 2019.

[47] Joshua Hochuli, Alec Helbling, Tamar Skaist, Matthew Ragoza, and David Ryan Koes. Visualizing convolutional neural network protein-ligand scoring. *Journal of molecular graphics & modelling*, 84:96–108, sep 2018.

[48] Narayanan Eswar, Ben Webb, Marc A Marti-Renom, M S Madhusudhan, David Eramian, Min-Yi Shen, Ursula Pieper, and Andrej Sali. Comparative protein structure modeling using Modeller. *Current Protocols in Bioinformatics*, Chapter 5:Unit 5.6, oct 2006.

[49] Georgii G Krivov, Maxim V Shapovalov, and Roland L Dunbrack. Improved prediction of protein side-chain conformations with SCWRL4. *Proteins*, 77(4):778–795, dec 2009.

[50] Todd J Dolinsky, Paul Czodrowski, Hui Li, Jens E Nielsen, Jan H Jensen, Gerhard Klebe, and Nathan A Baker. PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations. *Nucleic Acids Research*, 35(Web Server issue):W522–5, jul 2007.

[51] W Kabsch and C Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, dec 1983.

[52] John Vivian, Arjun Arkal Rao, Frank Austin Nothaft, Christopher Ketchum, Joel Armstrong, Adam Novak, Jacob Pfeil, Jake Narkizian, Alden D Deran, Audrey Musselman-Brown, Hannes Schmidt, Peter Amstutz, Brian Craft, Mary Goldman, Kate Rosenbloom, Melissa Cline, Brian O'Connor, Megan Hanna, Chet Birger, W James Kent, David A Patterson, Anthony D Joseph, Jingchun Zhu, Sasha Zaranek, Gad Getz, David Haussler, and Benedict Paten. Toil enables reproducible, open source, big biomedical data analyses. *Nature Biotechnology*, 35(4):314–316, apr 2017.

[53] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[54] G. W. Stewart. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal on Numerical Analysis*, 17(3):403–409, 1980.

[55] JunYoung Gwak, Christopher B Choy, and Silvio Savarese. Generative sparse detection networks for 3d single-shot object detection. In *European conference on computer vision*, 2020.

[56] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv*, dec 2013.

[57] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[58] William Falcon, Jirka Borovec, Adrian Wälchli, Nic Eggert, Justus Schock, Jeremy Jordan, Nicki Skafte, Ir1dXD, Vadim Bereznyuk, Ethan Harris, Tullie Murrell, Peter Yu, Sebastian Præsius, Travis Addair, Jacob Zhong, Dmitry Lipin, So Uchida, Shreyas Bapat, Hendrik Schröter, Boris Dayma, Alexey Karnachev, Akshay Kulkarni, Shunta Komatsu, Martin.B, Jean-Baptiste SCHIRATTI, Hadrien Mary, Donal Byrne, Cristobal Eyzaguirre, cinjon, and Anton Bakhtin. Pytorchlightning/pytorch-lightning: 0.7.6 release, May 2020.

[59] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com.

[60] Robert C Edgar. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, 26(19):2460–2461, oct 2010.

[61] Jaina Mistry, Robert D Finn, Sean R Eddy, Alex Bateman, and Marco Punta. Challenges in homology search: HMMER3 and convergent evolution of coiled-coil regions. *Nucleic Acids Research*, 41(12):e121, jul 2013.

[62] Robert C Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, mar 2004.

[63] Thomas A Hopf, Anna G Green, Benjamin Schubert, Sophia Mersmann, Charlotta P I Schärfe, John B Ingraham, Agnes Toth-Petroczy, Kelly Brock, Adam J Riesselman, Perry Palmedo, Chan Kang, Robert Sheridan, Eli J Draizen, Christian Dallago, Chris Sander, and Debora S Marks. The EVcouplings python framework for coevolutionary sequence analysis. *Bioinformatics*, 35(9):1582–1584, may 2019.

[64] Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature Communications*, 12(1):2403, apr 2021.

[65] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences of the United States of America*, 118(15), apr 2021.

[66] Yang Zhang and Jeffrey Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Research*, 33(7):2302–2309, apr 2005.

[67] John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. *Advances in Neural Information Processing Systems*, 2019.

3.6 Supplemental Matrial

3.6.1 Superfamilies used in this paper

Table 3.1: CATH Superfamilies used in this study.

CATH Code	Name	Description	# Domains	# Representatives	Manual Urfold
1.10.10.10	Winged helix-like DNA-binding domain superfamily/Winged helix DNA-binding domain		3444	524	
1.10.238.10	EF-hand		1933	166	
1.10.490.10	Globins		2891	52	
1.10.510.10	Transferase (Phosphotransferase) domain 1		7219	148	
1.20.1260.10	Ferritin, core subunit, four-helix bundle	a major non-haem iron storage protein in animal, plants and microorganisms	2985	60	
2.30.30.100	SH3 type barrels	Includes Integrase, C-terminal domain superfamily, retroviral	1545	56	SBB
2.40.50.140	OB fold	Dihydrolipoamide Acetyltransferase, E2P Nucleic acid-binding proteins”	2879	227	SBB
2.60.40.10	Immunoglobulin		31905	873	
3.10.20.30	Beta-grasp domain	a core structure consisting of beta(2)-alpha-beta(2), which is similar to that found in ubiquitin.	520	48	beta-grasp (Ub)
3.30.1360.40	Gyrase A; domain 2		160	13	Sm-like ribonucleoproteins
3.30.1370.10	‘K Homology domain, type 1”		139	34	RRM/RBD(ish)
3.30.1380.10	Hedgehog domain		101	10	Sm-like ribonucleoproteins
3.30.230.10	Ribosomal Protein S5; domain 2		1274	45	Sm-like ribonucleoproteins
3.30.300.20	K homology (KH) domain	Could belong together w/ the RRMs in a new Urfold	529	30	RRM/RBD(ish)
3.30.310.60	Sm-like ribonucleoprotein	C-terminal domain	28	1	Sm-like ribonucleoproteins
3.40.50.300	P-loop NTPases	P-loop containing nucleotide triphosphate hydrolases (both 2ak3 and 1reb in same cath id)	9233	561	P-loop NTPases
3.40.50.720	NAD(P)-binding Rossmann-like Domain		11728	647	Rossmann-based
3.80.10.10	Ribonuclease Inhibitor	Luecine rich repeat protein; positive control, sanity check	709	99	
3.90.420.10	‘Oxidoreductase, molybdopterin-binding domain”		58	6	beta-grasp (Ub)
3.90.79.10	NTP Pyrophosphohydrolase		850	74	beta-grasp (Ub)

3.6.2 Voxelization & Featurization

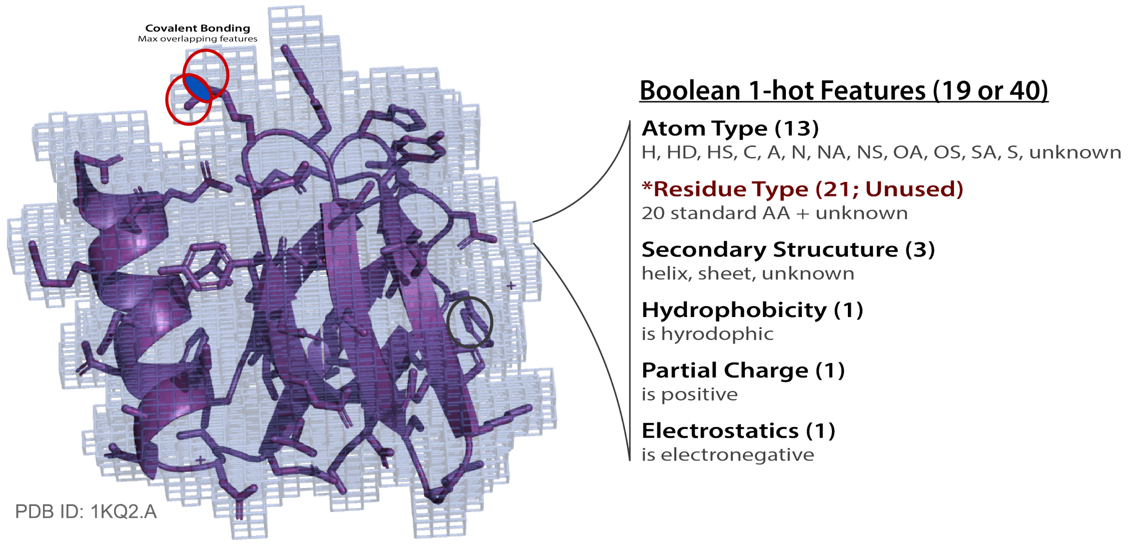


Figure 3.6: Voxelization & Featurization Method. Each domain is voxelized by (1) centering in a 256 ^{3} volume; and (2) discretizing each atom to fit 1 ^{3} voxels searching a KD-Tree with a radius equal to the current atoms van der walls radii, where the KD-tree initialized by the full 256 ^{3} volume, with 1 ^{3} resolution. If two or more atoms occupy a single voxel, the maximum from each feature is used to handle covalent bonding. Each voxel contains a 1-hot feature vector with 19 or 40 features depending on if residue type is included. Residue type was not included in the models shown in this paper because due poor reconstruction metrics so was not considered useful for this type of model, seen in Fig 3.8

3.6.3 Immunoglobulin (2.60.40.10) Model Metrics

Training & Validation Loss

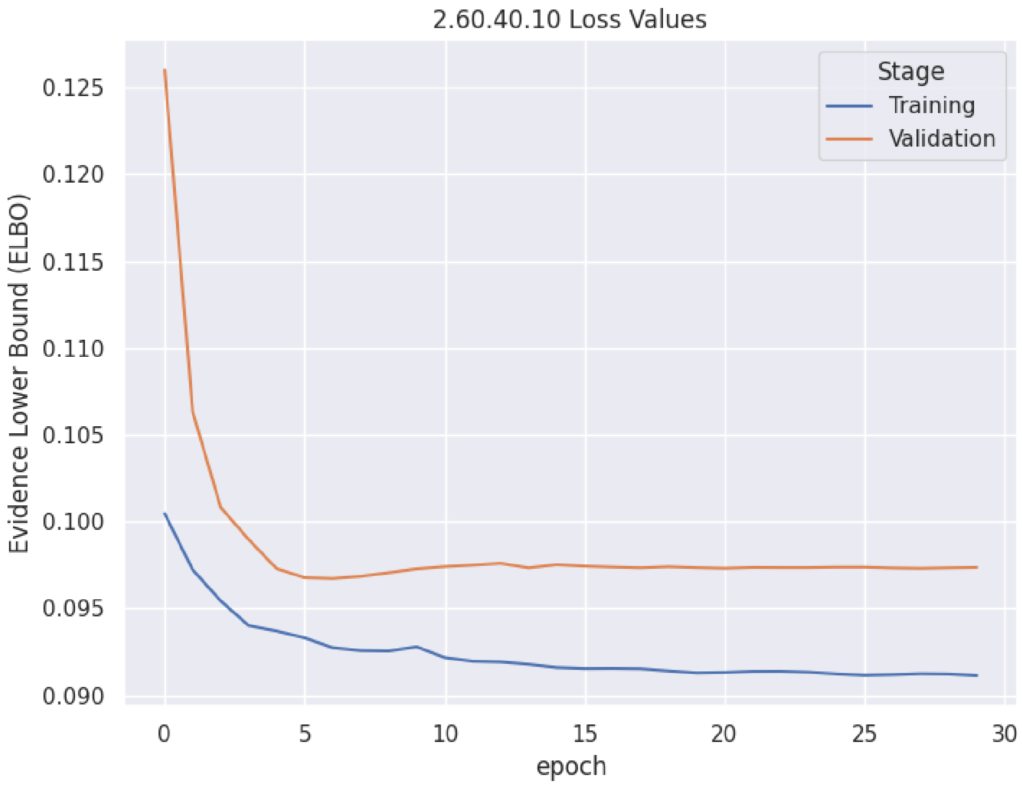


Figure 3.7: The 2.60.40.10 model was trained for 30 epochs using a 80% / 10 % split from CATH's S35 clusters (test [10 %] not shown).

Classification Metrics (w/ Residue Type)

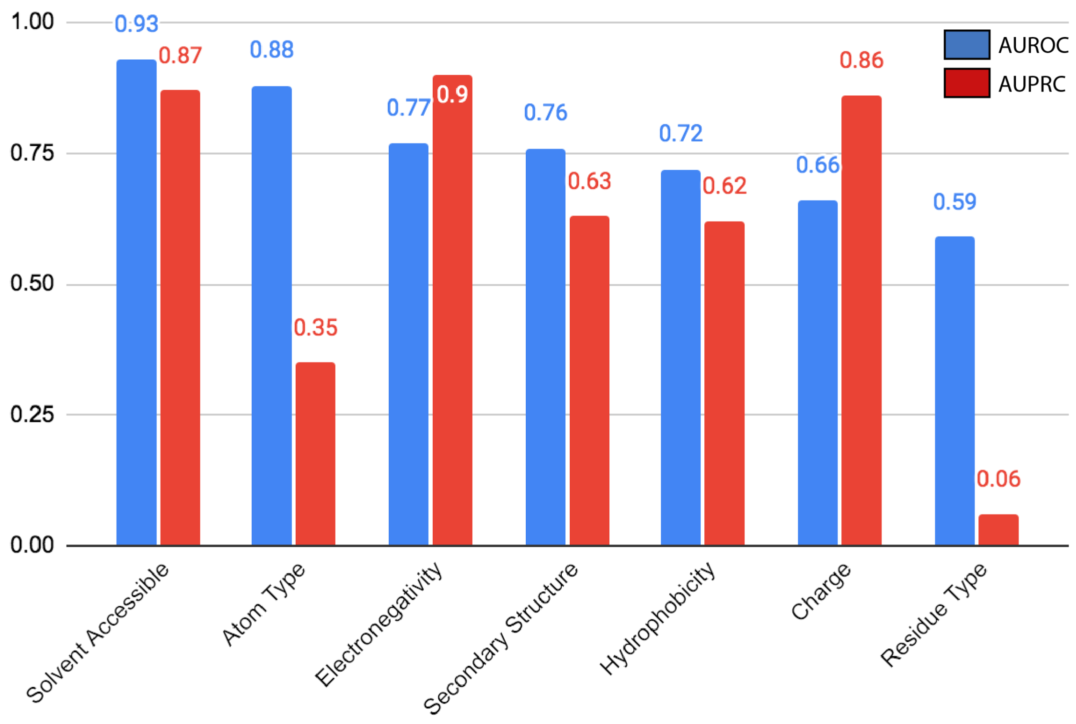


Figure 3.8: Classification Metrics for 7 Different Groups of Features Including Residue Type. We trained an Immunoglobulin-specific model 7 different times for 7 different groups of features, excluding all of the other feature groups. We compared the reconstructed values for each values to the input using ROC (receiver operating; true positive vs false positive) and PRC (precision vs recall) curves, saving the AUC (area under curve) for each. Most features were able to be reconstructed well (AUC ≥ 0.6) except residue type so we removed them from further models. We hypothesize that residue type is not as important for atom-only models and is too coarse-grained to be meaningful for this context.

Classification Metrics (w/o Residue Type)

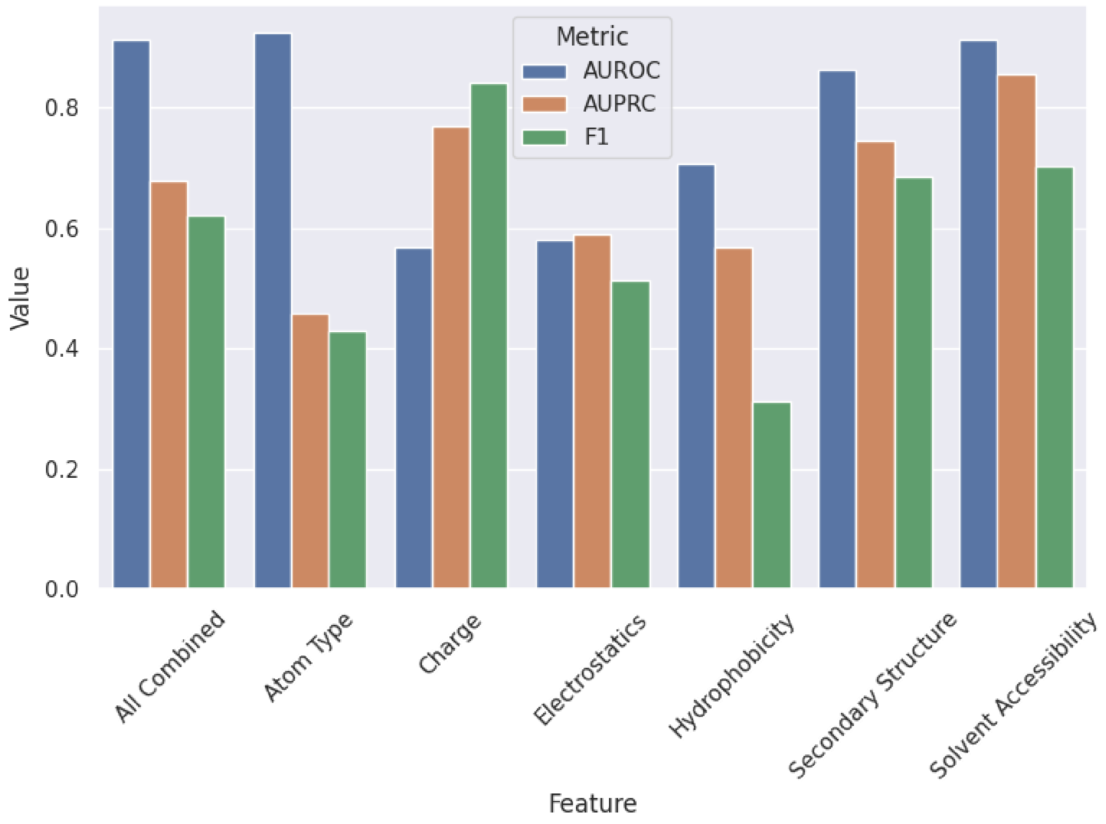
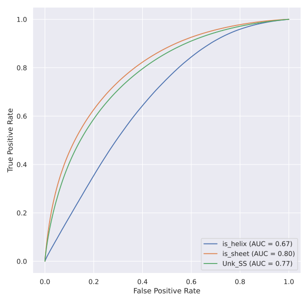
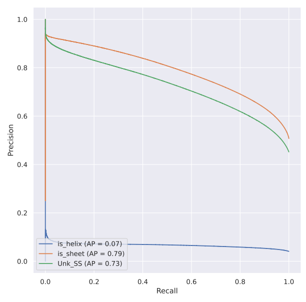


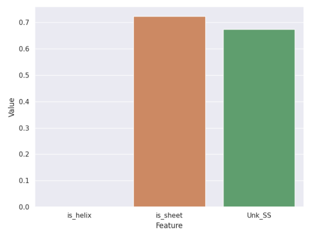
Figure 3.9: Classification Metrics for 7 Different Groups of Features Removing Residue Type. The 2.60.40.10 model was trained with all features, but individual feature groups were separated to perform micro-averaging ROC, PRC, and F1 scores



(a) Micro-averaged ROC Curve for separated features in the Secondary Structure feature group.

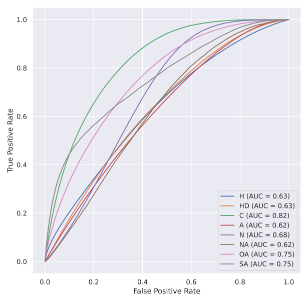


(b) Micro-averaged PRC Curve for separated features in the Secondary Structure feature group.

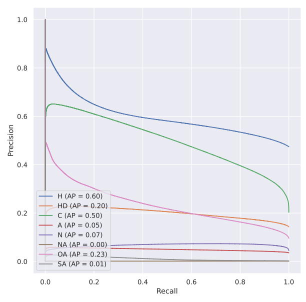


(c) Micro-averaged F1 values for separated features in the Secondary Structure feature group.

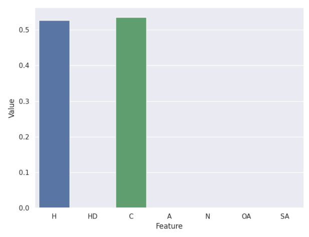
Figure 3.10: Classification metrics for separated features in the Secondary Structure feature group.



(a) Micro-averaged ROC Curve for separated features in the Atom Type feature group.



(b) Micro-averaged PRC Curve for separated features in the Atom Type feature group.

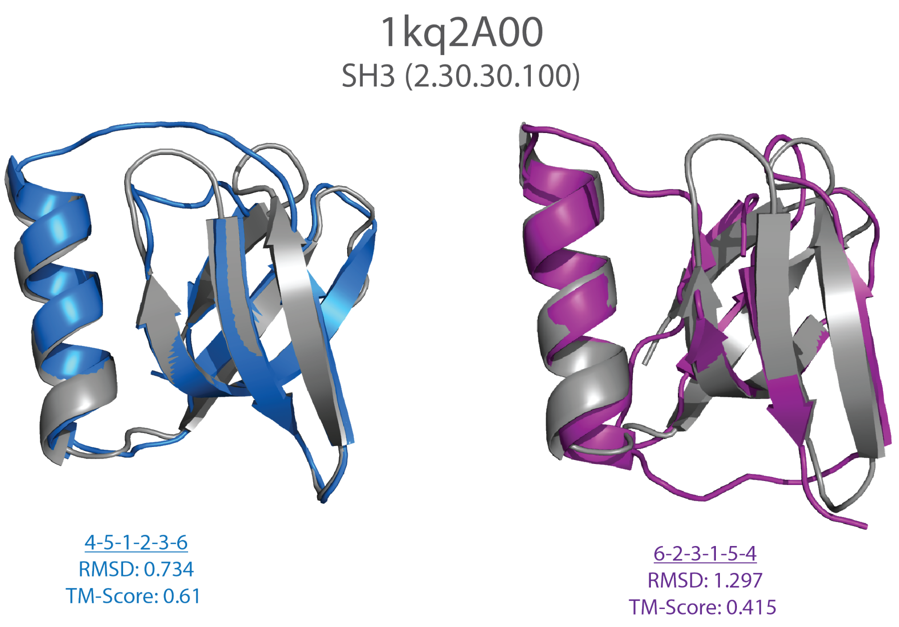


(c) Micro-averaged F1 values for separated features in the Atom Type feature group.

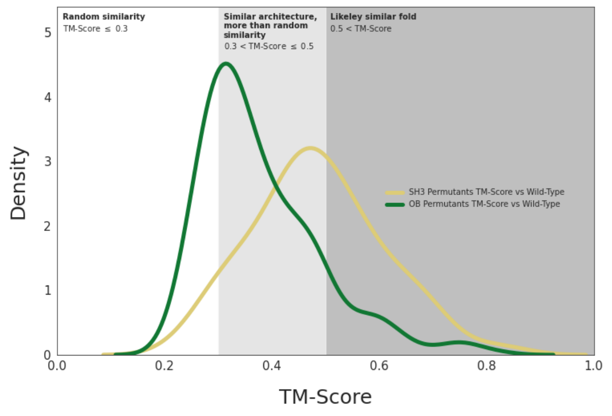
Figure 3.11: Classification metrics for separated features in the Atom Type feature group.

3.6.4 Multiple Loop Permutations

Permutants



(a) Exemplar SH3 permutatants



(b) TM-Scores for Multiple Loop Permuted structures. Many of the SH3 and OB permutant TM-Scores fall between 0.3-0.5, showing that that are more than randomly similar, but not the same fold.

Figure 3.12: Multiple Loop Permutations Example

Class Imbalance Scores

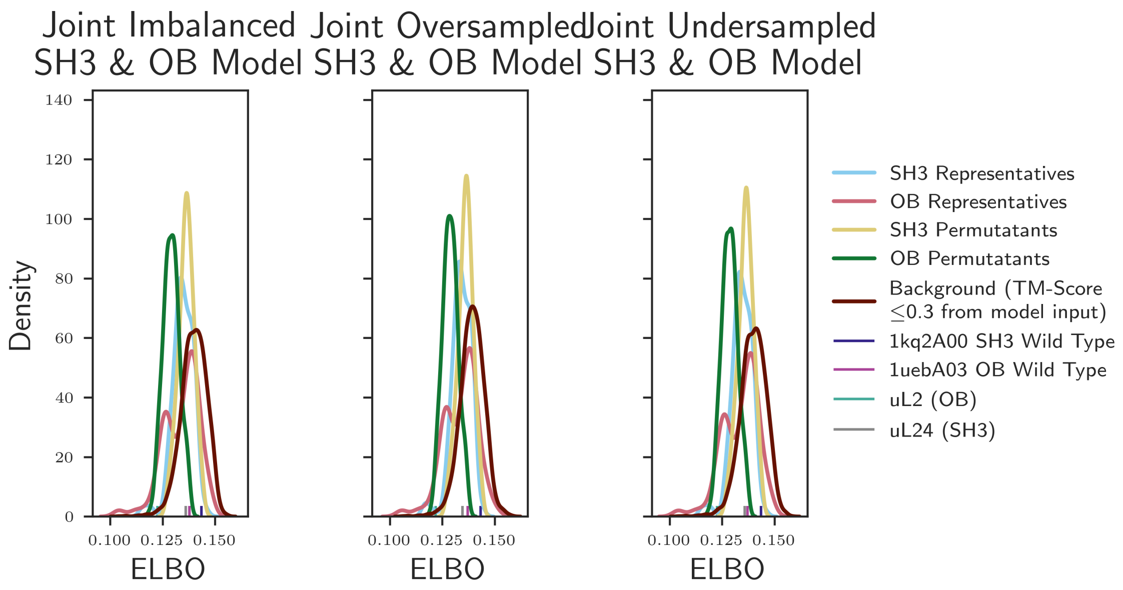
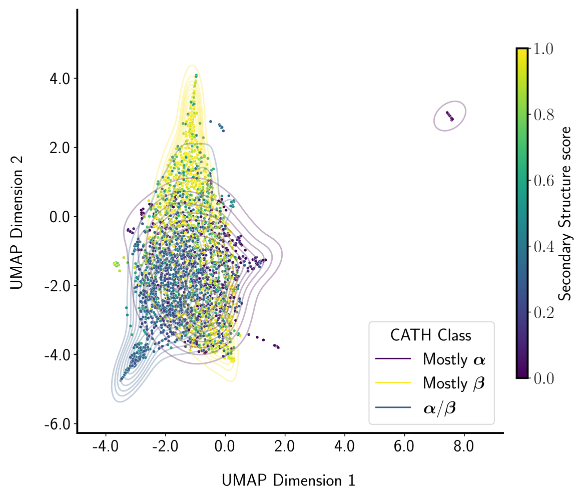


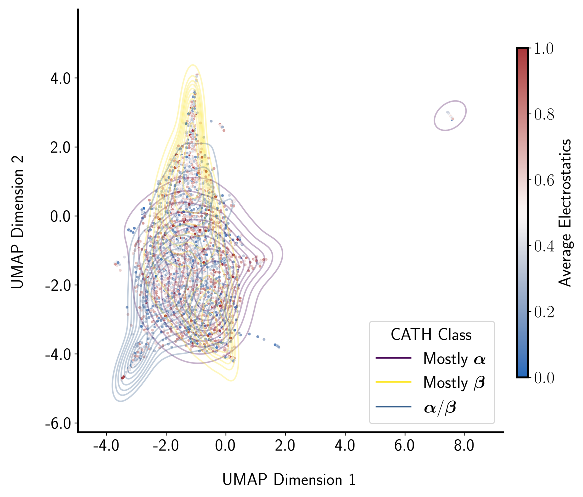
Figure 3.13: Class Imbalance Studied for SH3 and OB VAE models. In order test how class imbalance affects our models, we trained 3 joint SH3 and OB models: (A) using all domains from each superfamily; (B) oversampling SH3 domains to match the number of OB domains; and (3) under-sampling OB domains to match the number of SH3 domains. We found no significant change between them in terms of ELBO scores when running representatives, multi-loop permuted models, and ancestral versions of SH3 and OB.

3.6.5 Latent Space

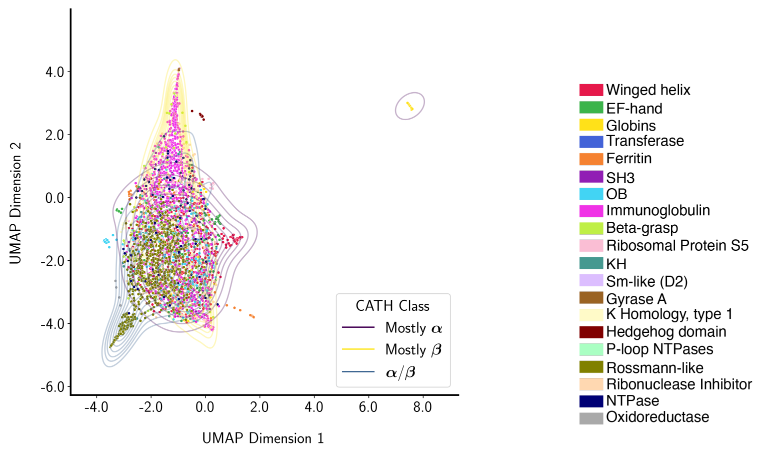
UMAP



(a) Colored by secondary structure



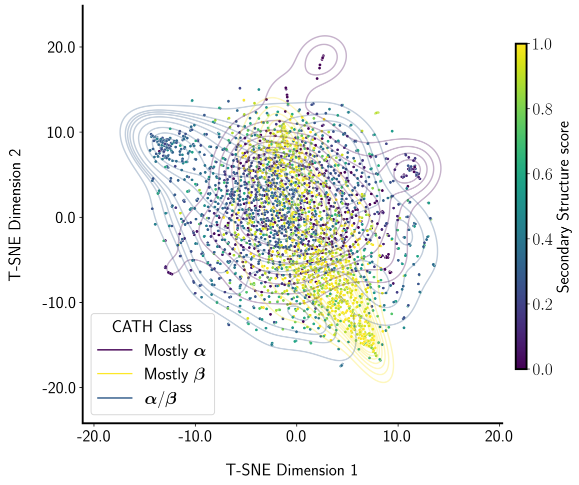
(b) Colored by averaged electro-negativity values



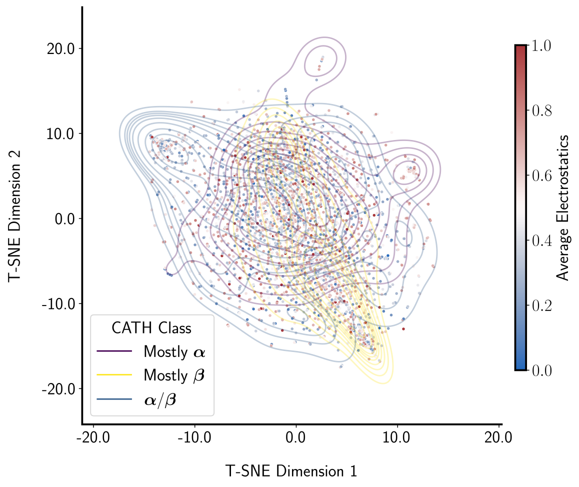
(c) Colored by true CATH Superfamily

Figure 3.14: Latent Space from UMAP. Representatives from each superfamily were subjected to models trained with domains from the same superfamily, saving the latent variable representing the mean for each representative domain. The latent variables for each different model were concatenated and reduced from 1024 dims to 2 for visualization

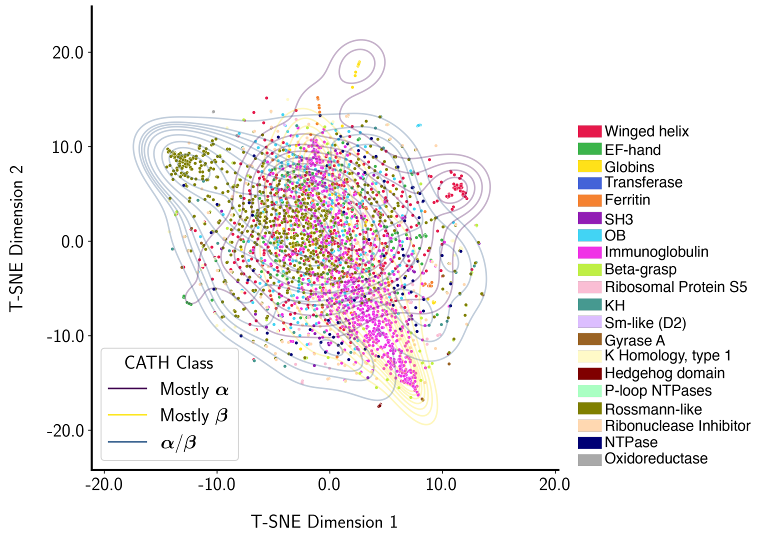
T-SNE



(a) Colored by secondary structure



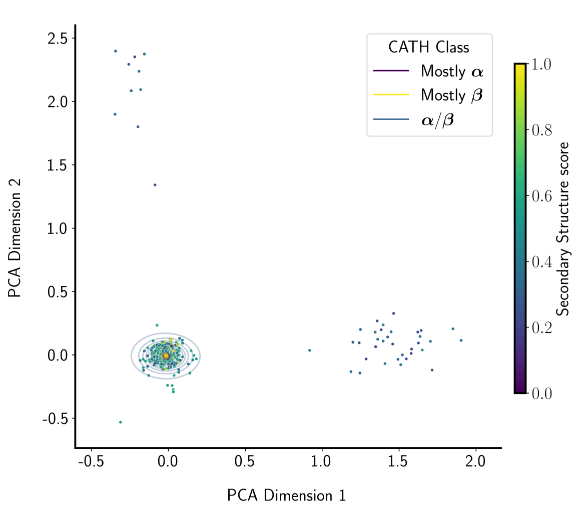
(b) Colored by averaged electro-negativity values



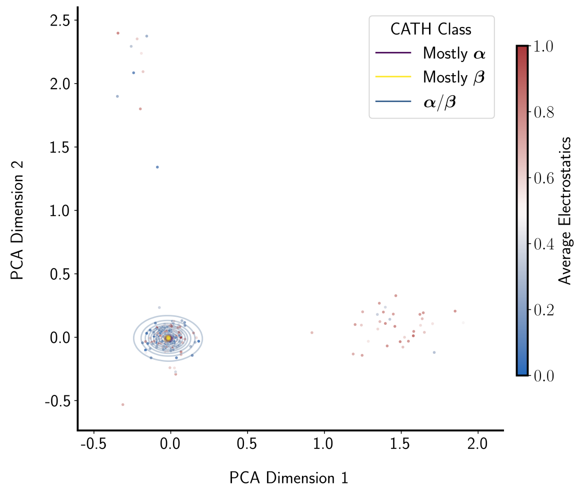
(c) Colored by true CATH Superfamily

Figure 3.15: Latent Space from T-SNE

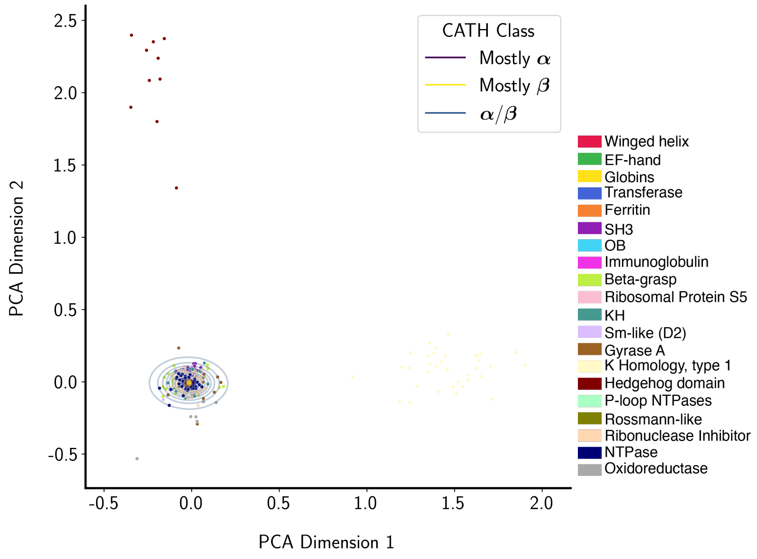
PCA



(a) Colored by secondary structure



(b) Colored by averaged electro-negativity values



(c) Colored by true CATH Superfamily

Figure 3.16: Latent Space from PCA

3.6.6 Stochastic Block Modelling

# of Clusters	20
Silhouette Score	−0.0705
Davies-Boundin Score	33.1678
Overlap Score	0.2798
Rand Score	0.8542
Rand Score Adjusted	0.1977
Adjusted Mutual Information	0.4108
Homogeneity Score	0.4831
Completeness Score	0.3749

Table 3.2: Clustering metrics of DeepUrfold SBM vs CATH

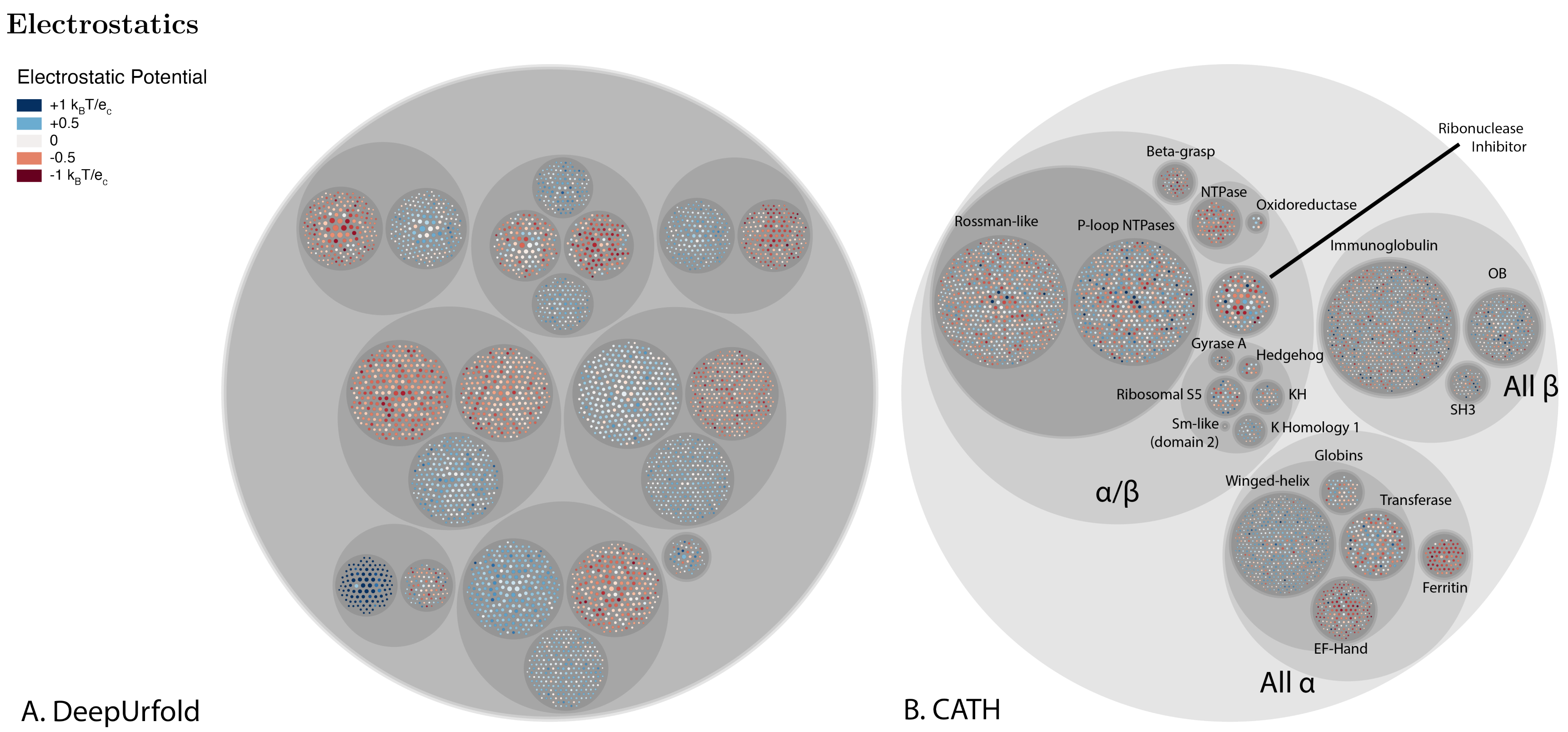


Figure 3.17: SBM Communities Colored by Electrostatic Potential. Each atom it annotated with the Boolean feature is_electronegative. We sum up all of the electronegtive atoms and take a fraction of the total number of atoms.

Charge

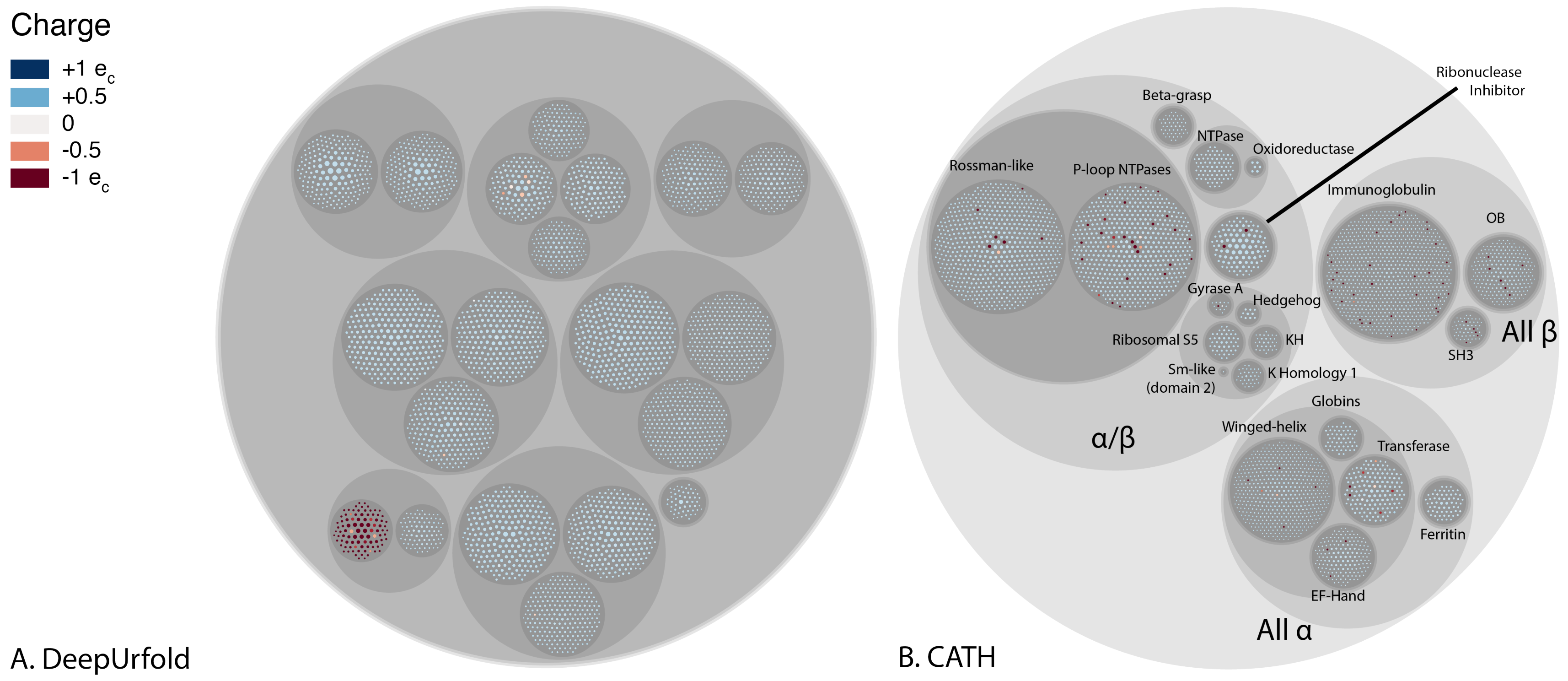


Figure 3.18: SBM Communities Colored by Partial Charge. Each atom it annotated with the Boolean feature is_positive. We sum up all of the positive atoms and take a fraction of the total number of atoms.

Secondary Structure

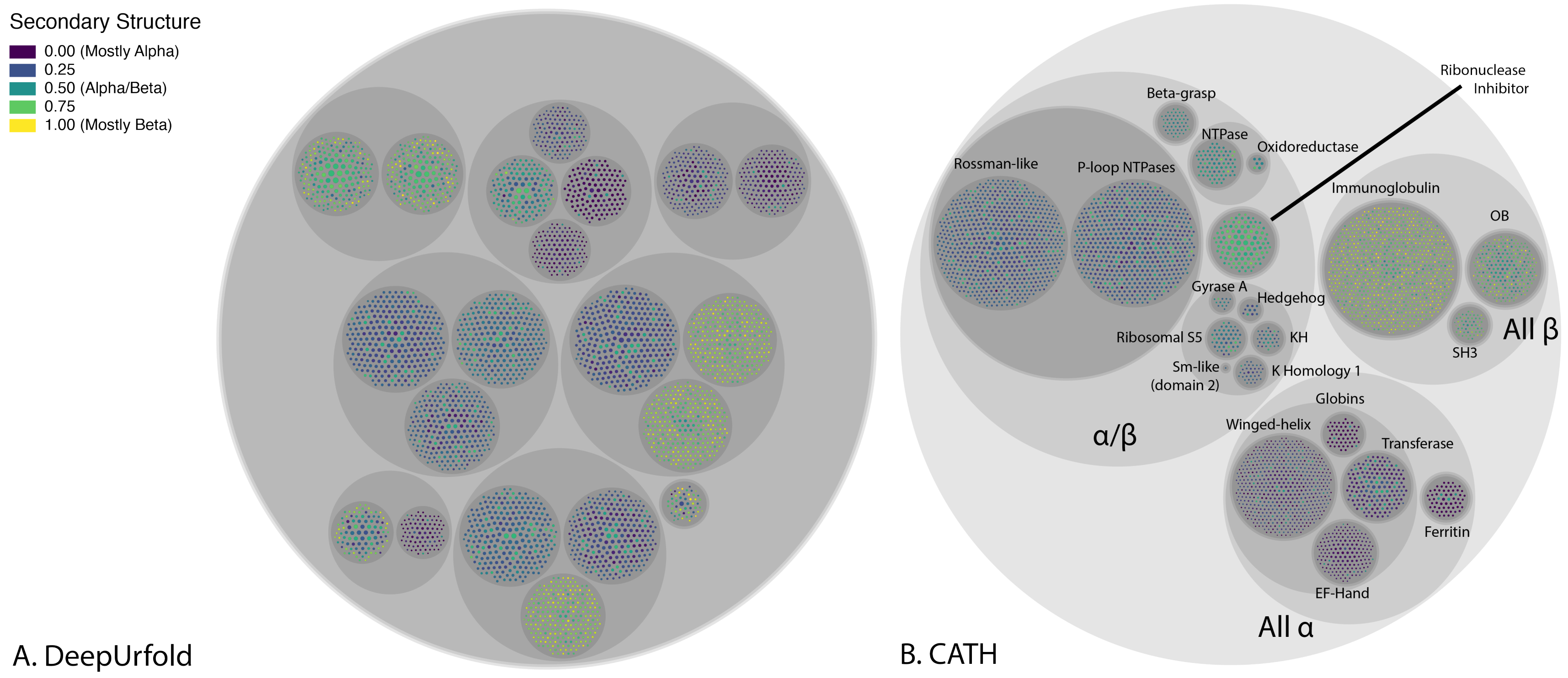


Figure 3.19: SBM Communities Colored by Secondary Structure. To calculate the secondary structure score, we use the formula: (# beta atoms - # alpha atoms)/(2*(# beta atoms + # alpha atoms)) + 0.5.

GO: Molecular Function

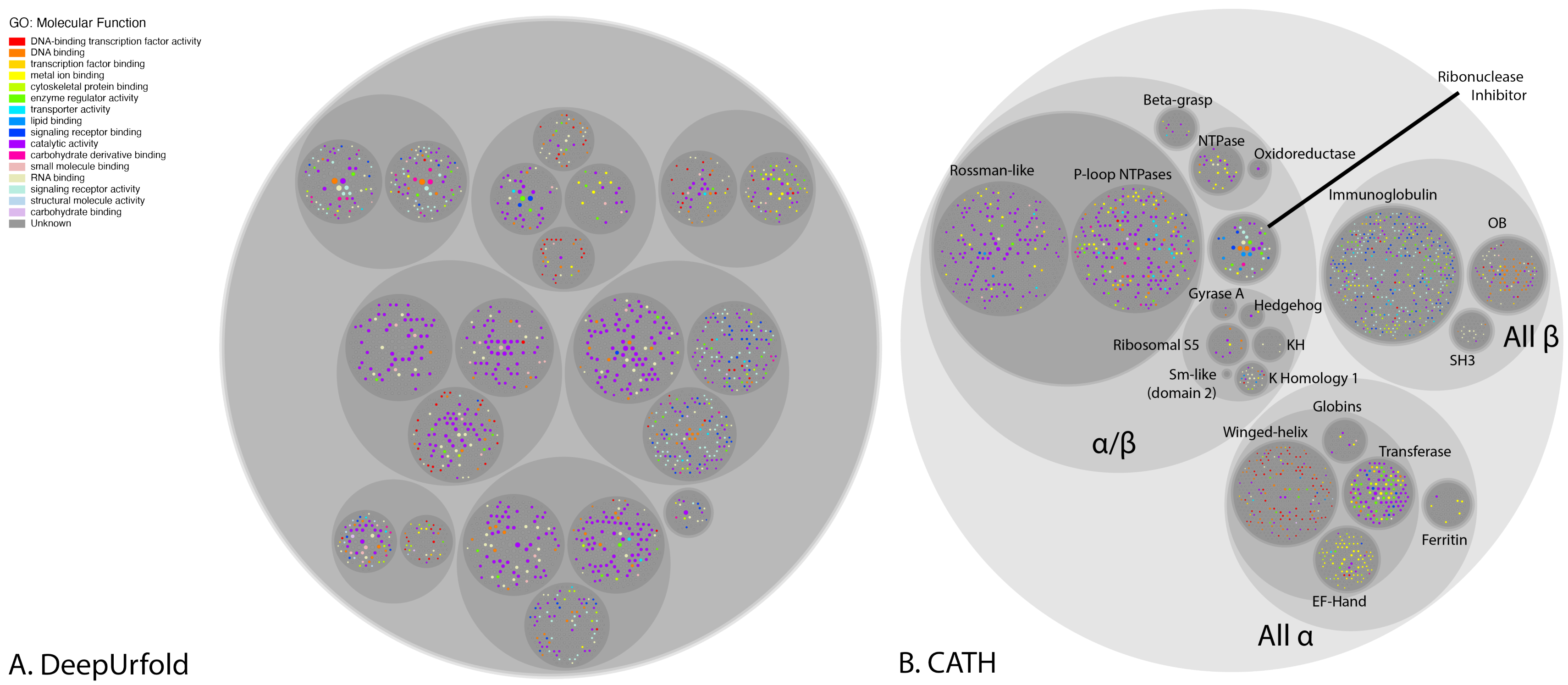


Figure 3.20: SBM Communtities Colored by GO MF enrichment. We use GOATOOLS to calculate enrichment for each GO term from all domains in the predicted SBM community (leaf grouping only) using GO Slim terms from AGR. If the domain has a GO term that is enriched in its community (p_fdr_bh $\leq$ 0.05), then it is colored for the associated term. If there are multiple enriched terms, only the first is used.

GO: Biological Process

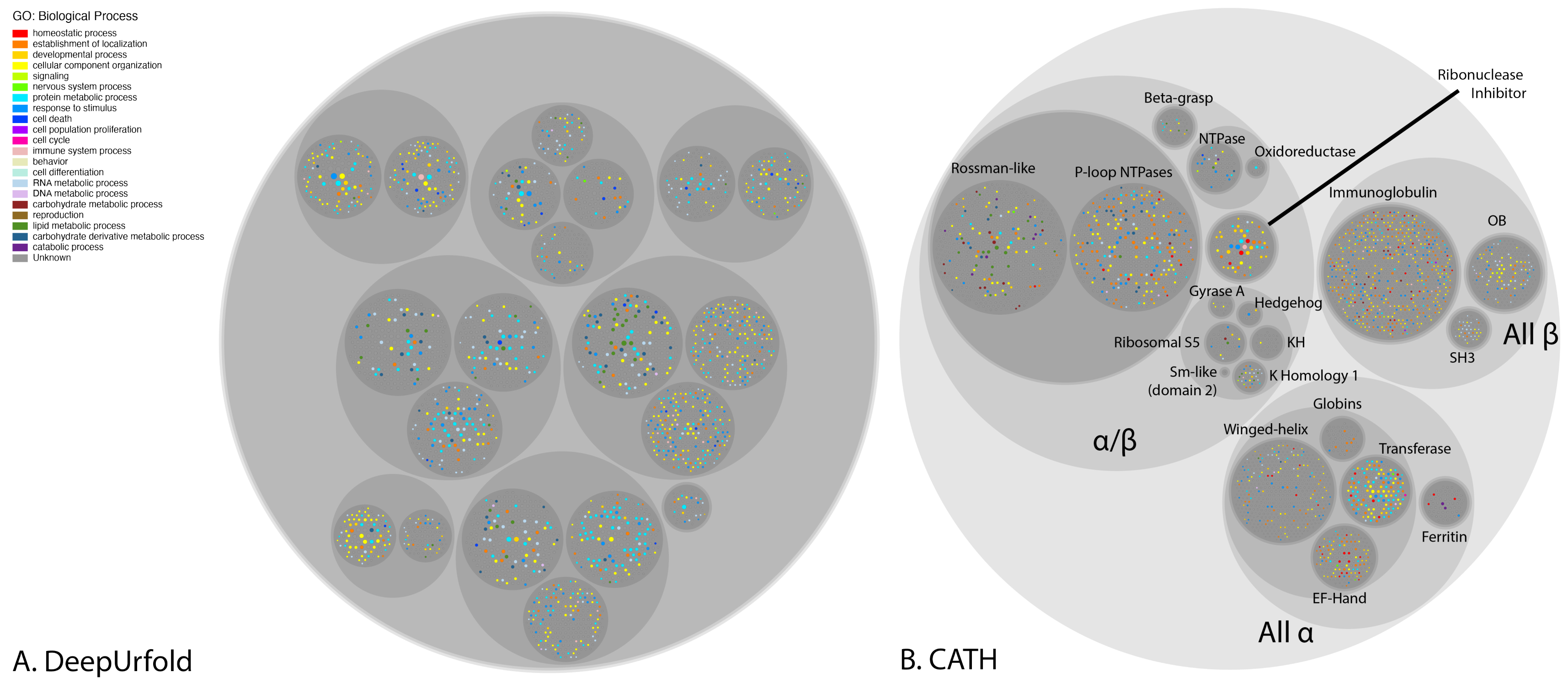


Figure 3.21: SBM Communtities Colored by GO BP enrichment. We use GOATOOLS to calculate enrichment for each GO term from all domains in the predicted SBM community (leaf grouping only) using GO Slim terms from AGR. If the domain has a GO term that is enriched in its community (p_fdr_bh ≤ 0.05), then it is colored for the associated term. If there are multiple enriched terms, only the first is used.

GO: Cellular Component

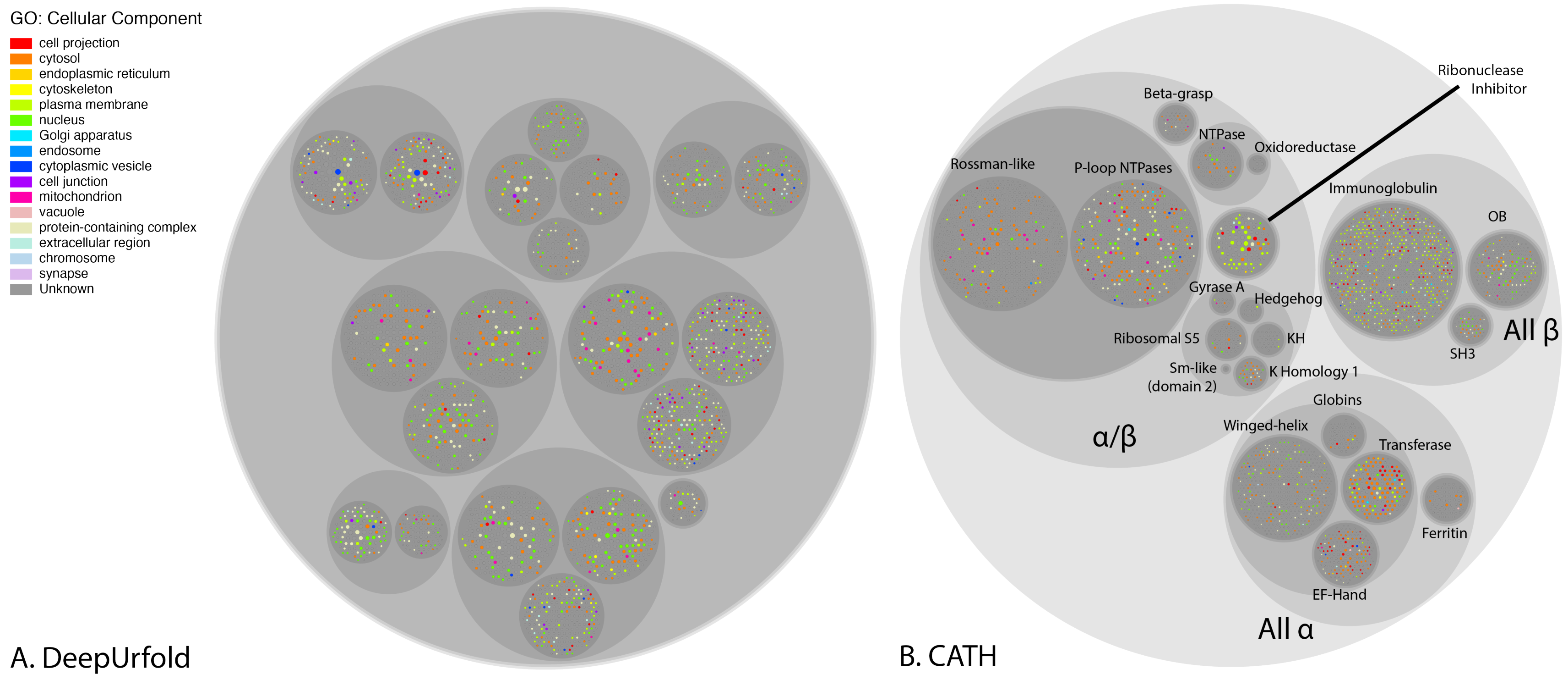


Figure 3.22: SBM Communtities Colored by GO CC enrichment. We use GOATOOLS to calculate enrichment for each GO term from all domains in the predicted SBM community (leaf grouping only) using GO Slim terms from AGR. If the domain has a GO term that is enriched in its community (p_fdr_bh ≤ 0.05), then it is colored for the associated term. If there are multiple enriched terms, only the first is used.

Downsampled SBM

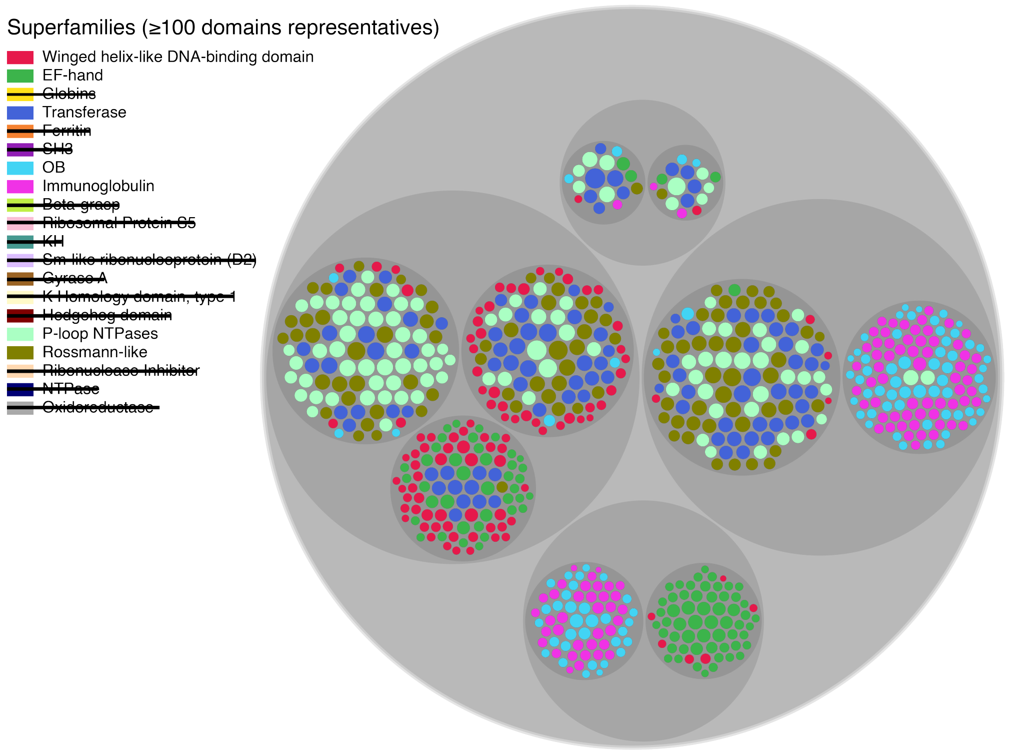


Figure 3.23: Class Imbalance Studies during Stochastic Block Modelling. In order to test how the SBM treats highly imbalanced classes, we included only superfamilies that had $\geq$ 100 domain representatives and sampled 100 random domains from each. No immediate change can be detected and OB domains are still found in the same community as Immunoglobulins

3.6.7 Comparison to Other metrics

Table 3.3: Comparison of Stochastic Block Modelling of a Bipartite graph of CATH Domains and Superfamilies with scores based on similar algorithms to DeepUrfold. Silhouette Score and Davis-Boundin are self measures and do not compare against CATH.

	Method	Model	Train Data	Score	**# Communities**	*Silhouette Score*	*Davis-Boundin*	*Overlap*	*Rand Score (Adjusted)*	*Adjusted Mutual Information*	*Homogeneity*	*Completeness*
Sequence-based												
Pairwise	UCLUST	Global Alignment	CATH S35	↑% Seq ID	19	0.2863	2.4830	0.6148	0.9131 (0.6778)	0.7703	0.7712	0.7791
		Local Alignment	CATH S35	↑% Seq ID	38	−0.2584	10.4408	0.3476	0.8338 (0.2730)	0.4382	0.5074	0.4156
Single Model	ESM-1b	Language Transformer	Uniref50	↓Euclidean Distance	121	0.4299	0.7931	0.1676	0.8638 (0.0896)	0.5843	0.9065	0.4311
Superfamily-specific models	HMMER	HMM	CATH S35	↑Bitscore	24	−0.1715	5.3797	0.4122	0.7463 (0.2444)	0.4747	0.4429	0.5416
			EVcouplings alignment	↑Bitscore	14	0.5422	3.5902	0.3105	0.5546 (0.0905)	0.2173	0.1734	0.3382
	SeqDesign	Auto-regressive	EVcouplings alignment (Unaligned)	↑Bitscore	7	0.0410	3.6629	0.4095	0.7604 (0.2085)	0.2736	0.2370	0.3235
Structure-based												
Pairwise	TM-Align	Structural Alignment	CATH S35	↑TM-Score	42	0.1303	1.7138	0.3150	0.8734 (0.2380)	0.6232	0.8247	0.5188
				↓RMSD	29	0.1593	1.8141	0.4200	0.8831 (0.3342)	0.6628	0.8172	0.5706
		Circular Permutations	CATH S35	↑TM-Score	42	0.1124	1.7407	0.2969	0.8702 (0.2162)	0.6042	0.8005	0.5037
				↓RMSD	26	0.1801	1.6529	0.4164	0.8799 (0.3467)	0.6740	0.8032	0.5929
Superfamily-specific models	Struct2Seq	Graph Transformer	CATH (Backbone)	↑Perplexity	15	0.0563	2.5657	0.4724	0.8745 (0.3959)	0.5434	0.5857	0.5196
	DeepUrfold (ours)	3D-CNN VAE	CATH (Full)	↓ELBO	20	−0.0705	33.1678	0.2798	0.8542 (0.1977)	0.4108	0.4831	0.3749

3.6.8 Model Architecture

```
DomainStructureVAE(
  (encoder): Encoder(
    (block1): Sequential(
      (0): MinkowskiConvolution(in=20, out=16, kernel_size=[3, 3, 3],
         stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(16, eps=1e-05, momentum=0.1, affine=
         True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=16, out=16, kernel_size=[3, 3, 3],
         stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(16, eps=1e-05, momentum=0.1, affine=
         True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block2): Sequential(
      (0): MinkowskiConvolution(in=16, out=32, kernel_size=[3, 3, 3],
         stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(32, eps=1e-05, momentum=0.1, affine=
         True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=32, out=32, kernel_size=[3, 3, 3],
         stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(32, eps=1e-05, momentum=0.1, affine=
         True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block3): Sequential(
      (0): MinkowskiConvolution(in=32, out=64, kernel_size=[3, 3, 3],
         stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(64, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=64, out=64, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(64, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block4): Sequential(
      (0): MinkowskiConvolution(in=64, out=128, kernel_size=[3, 3, 3],
          stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(128, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=128, out=128, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(128, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block5): Sequential(
      (0): MinkowskiConvolution(in=128, out=256, kernel_size=[3, 3, 3],
          stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(256, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=256, out=256, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(256, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block6): Sequential(
      (0): MinkowskiConvolution(in=256, out=512, kernel_size=[3, 3, 3],
          stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(512, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=512, out=512, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(512, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block7): Sequential(
      (0): MinkowskiConvolution(in=512, out=1024, kernel_size=[3, 3, 3],
           stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(1024, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=1024, out=1024, kernel_size=[3, 3,
          3], stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(1024, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (global_pool): MinkowskiGlobalPooling(mode=PoolingMode.
        GLOBAL_AVG_POOLING_PYTORCH_INDEX)
    (linear_mean): MinkowskiLinear(in_features=1024, out_features=1024,
        bias=True)
    (linear_log_var): MinkowskiLinear(in_features=1024, out_features
        =1024, bias=True)
  )
  (decoder): Decoder(
    (block1): Sequential(
      (0): MinkowskiConvolutionTranspose(in=1024, out=1024, kernel_size
          =[2, 2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(1024, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=1024, out=1024, kernel_size=[3, 3,
          3], stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(1024, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
      (6): MinkowskiConvolutionTranspose(in=1024, out=512, kernel_size
          =[2, 2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (7): MinkowskiSyncBatchNorm(512, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (8): MinkowskiELU()
      (9): MinkowskiConvolution(in=512, out=512, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (10): MinkowskiSyncBatchNorm(512, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (11): MinkowskiELU()
    )
    (block2): Sequential(
      (0): MinkowskiConvolutionTranspose(in=512, out=256, kernel_size
          =[2, 2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(256, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=256, out=256, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(256, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block3): Sequential(
      (0): MinkowskiConvolutionTranspose(in=256, out=128, kernel_size
          =[2, 2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(128, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=128, out=128, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(128, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block4): Sequential(
      (0): MinkowskiConvolutionTranspose(in=128, out=64, kernel_size=[2,
           2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(64, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=64, out=64, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(64, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block5): Sequential(
      (0): MinkowskiConvolutionTranspose(in=64, out=32, kernel_size=[2,
          2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(32, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=32, out=32, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(32, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block6): Sequential(
      (0): MinkowskiConvolutionTranspose(in=32, out=16, kernel_size=[2,
          2, 2], stride=[2, 2, 2], dilation=[1, 1, 1])
      (1): MinkowskiSyncBatchNorm(16, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (2): MinkowskiELU()
      (3): MinkowskiConvolution(in=16, out=16, kernel_size=[3, 3, 3],
          stride=[1, 1, 1], dilation=[1, 1, 1])
      (4): MinkowskiSyncBatchNorm(16, eps=1e-05, momentum=0.1, affine=
          True, track_running_stats=True)
      (5): MinkowskiELU()
    )
    (block7): MinkowskiConvolution(in=16, out=20, kernel_size=[1, 1, 1],
        stride=[1, 1, 1], dilation=[1, 1, 1])
    (pruning): MinkowskiPruning()
  )
)
```

Chapter 4

DeepUrfold-explain: Explainable Deep Generative Models, Ancestral Fragments, and Murky Regions of the Protein Structure Universe

Eli J. Draizen[1,2], Cameron Mura[2], and Philip E. Bourne[1,2]

[1] Department of Biomedical Engineering, University of Virginia, Charlottesville, VA 22903

[2] School of Data Science, University of Virginia, Charlottesville, VA 22903

Accepted to Machine Learning for Structural Biology Workshop, NeurIPS 2022 and is available as a preprint at https://doi.org/10.1101/2022.11.16.516787.

Abstract

Modern proteins did not arise abruptly, as singular events, but rather over the course of at least 3.5 billion years of evolution. Can machine learning teach us how this occurred? The molecular evolutionary processes that yielded the intricate three-dimensional (3D) structures of proteins involve duplication, recombination and mutation of genetic elements, corresponding to short peptide fragments. Identifying and elucidating these ancestral fragments is crucial to deciphering the interrelationships amongst proteins, as well as how evolution acts upon protein sequences, structures & functions. Traditionally, structural fragments have been found using sequence and 3D structural alignment, but that becomes challenging when proteins have undergone extensive permutations—allowing two proteins to share a common architecture, though their topologies may drastically differ (a phenomenon termed the *Urfold*). We have designed a new framework to identify compact, potentially-discontinuous peptide fragments by combining (i) deep generative models of protein superfamilies with (ii) layer-wise relevance propagation (LRP) to identify atoms of great relevance in creating an embedding during an $\text{all}_{superfamilies} \times \text{all}_{domains}$ analysis. Our approach recapitulates known relationships amongst the evolutionarily ancient small β-barrels (e.g. SH3 and OB folds) and P-loop–containing proteins (e.g. Rossmann and P-loop NTPases), previously established via manual analysis. Because of the generality of our deep model's approach, we anticipate that it can enable the discovery of new ancestral peptides. In a sense, our framework uses LRP as an 'explainable AI' approach, in conjunction with a recent deep generative model of protein structure (termed *Deep-Urfold*), in order to leverage decades worth of structural biology knowledge to decipher the underlying molecular bases for protein structural relationships—including those which are exceedingly remote, yet can be discovered via deep learning.

4.1 Introduction

Historically, protein structural evolution has been studied via painstaking visual inspection and manual analyses of structures, including a heavy reliance on comparisons built upon 3D superposition/alignment of atomic coordinates. Indeed, visualizing protein structures and their 3D alignments using graphical tools such as 'ribbon diagrams' has enabled many landmark discoveries in biology and medicine, largely because these diagrams simplify the inordinately complicated geometric structures of proteins into something that is comprehensible by the human brain. However, simplified cartoon representations ignore the high-dimensional, largely biophysical/physicochemical feature space of all the atoms defining a protein, and an over-reliance on simplified, static representations can limit us to seeing *a* structure of a particular protein as being *the* structure—i.e., we fall prey to viewing as important only one particular geometric representation or structural conformation, versus the true statistical ensemble of thermally-accesible states that an actual protein structure samples in reality. In short, conventional schemes for conceptualizing and analyzing protein structure relationships are not without limitations, and can cause us to miss phylogenetically remote (deeply ancestral) relationships. Some have referred to this pitfall as the 'curse of the ribbon' [1]. Modern deep learning–based methods enable fundamentally new representations of proteins and their sequence/structure/function relationships—for example, as lower-dimensional embeddings that incorporate biophysical properties (e.g., electrostatics) alongside 3D-coordinate data (i.e., geometry), sequence information and residue profiles, and so on. All of this, in turn, might finally lift the curse.

While many new sequence-based deep learning methods, based on large language models [2, 3, 4, 5], can identify more remote similarities than can hidden Markov models (HMMs) or classic sequence-comparison algorithms (e.g., BLAST), larger structural rearrangements and permutations, such as occur on evolutionary timescales, are still difficult to detect. If one views the protein universe through the lens of a hierarchical classification scheme such as CATH [6], most new homologous sequences identified by these methods would be located within the same homologous superfamily or topology strata—i.e., there are bounds on how remote a homology can be detected by existing methods. Thus, sets of potentially distantly-related proteins, with similar architectures yet different topologies, are generally missed [7]. Indeed, it was recently proposed that there might exist a new level of structural granularity, lying between the architecture and topology levels (the latter of which is synonymous with a given protein's *fold*); termed the 'Urfold', this provisional new level would naturally allow for 3D fragments that are spatially compact yet potentially discontinuous in sequence [7], such as may be the case with ancient, deeply ancestral fragments. An example of an urfold can be seen in Fig 4.1, highlighting two widespread classes of phosphate-binding loop (PBL)–containing proteins, namely the Rossmann fold-containing proteins and P-loop NTPases.

The Urfold model of protein structure—and, thus, protein structural interrelationships—arose by noticing the striking structural and functional similarities among deeply-divergent collections of domains from the SH3 and OB superfamilies [8]. Those superfamilies all contain a small β-barrel (SBB) domain, composed of five β-strands, but the topologies/connections between the strands have been permuted such that these proteins often share less than 20% sequence identity between one another (i.e., below the classic 'twilight zone' of similarity for inferring homology between two sequences). Despite having permuted strands, the architecturally-identical SBBs are often involved in nucleic acid metabolic pathways, and many of them oligomerize via residue interactions amongst similar edge-strands [8]. Recently, it was proposed that the SH3 and OB share a common ancestor that diverged via a process called ‘Creative Destruction’ [9, 10]. Notably, the SH3 and OB are two of the most ancient and widespread protein folds, and they permeate most information-storage and information-processing pathways in cellular life, from DNA replication to transcription of DNA→RNA and translation of RNA→protein [9, 11].

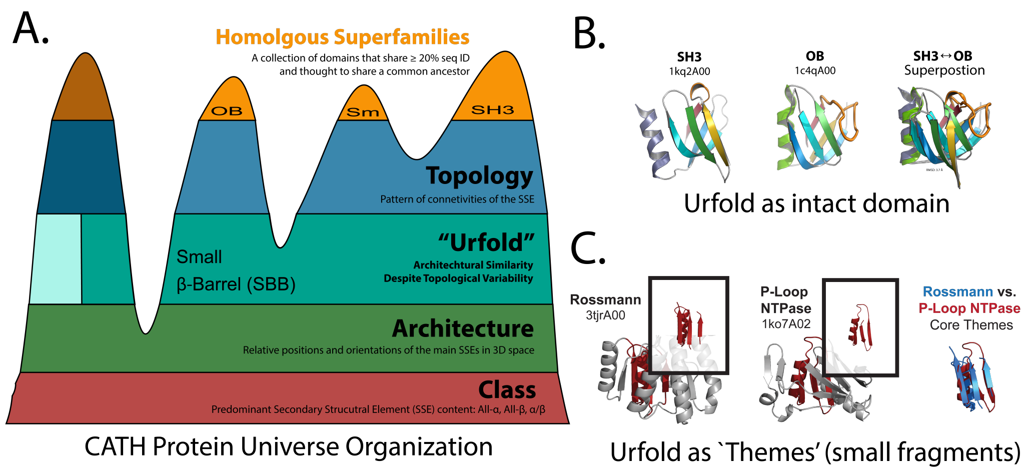

A.
Homolgous Superfamilies
A collection of domains that share ≥ 20% seq ID and thought to share a common ancestor
OB
Sm
SH3
Topology
Pattern of connetivities of the SSE
"Urfold"
Architechtural Similarity Despite Topological Variability
Small β-Barrel (SBB)
Architecture
Relative positions and orientations of the main SSEs in 3D space
Class
Predominant Secondary Strucutral Element (SSE) content: All-α, All-β, α/β
CATH Protein Universe Organization
B.
SH3
1kq2A00
OB
1c4qA00
SH3↔OB
Superpostion
Urfold as intact domain
C.
Rossmann
3tjrA00
P-Loop
NTPase
1ko7A02
Rossmann vs.
P-Loop NTPase
Core Themes
Urfold as `Themes' (small fragments)

Figure 4.1: The Urfold Represents Architectural Similarity Despite Topological Variability. A) CATH hierarchically organizes the protein universe into Class, Architecture, Topology, and Homologous Superfamilies. We hypothesize the ‘Urfold’ strata would fall in between Architecture and Topology. B) SH3 and OB share a small β-barrel urfold. C) The Rossmann and P-loop NTPases both contain a common core theme of three β-strands connected to an α-helix by a glycine-rich loop; the loop is of functional importance by virtue of binding phosphorylated ribonucleoside ligands, be they substrates, cofactors, or otherwise. However, the Rossmann fold is permuted such that one β-strand is no longer nearby in sequence and another β-strand has reversed direction. These gross structural differences have led biologists to incorrectly classify these ancient folds as being unrelated. The two folds are now bridged via a ‘PBL’ urfold [12, 7].

Another prominent example of an urfold can be seen in the phosphate-binding loop (PBL) containing proteins, which include the Rossmann and P-loop NTPases superfamilies [12]. Both of these superfamilies contain a small protein fragment of three

β-strands that contact a single α-helix, with these structural elements linked by a phosphate-binding glycine-rich loop (Fig. 4.1). Most domains from the PBL urfold are quite large, i.e. featuring many additional secondary structural elements (SSEs) beyond the core, and they do not oligomerize; rather, they have a large central cavity for ligand-binding. While Rossmann and PBL proteins are known to bind a diverse repertoire of ligands and catalyze many different types of reactions, they predominately bind phosphorylated nucleotides and similar compounds, with the phosphate groups primarily interacting with the phosphate-binding loop [13].

In addition to similarity at the full-domain level, the Urfold model allows for subdomain–level structural fragments that may be discontinuous in sequence. That is, the Urfold extends a CATH-like hierarchical representational scheme of the protein universe by allowing for (conserved) spatial constellations of short peptides, perhaps like the *Ancestral Peptides* [14] or *Themes* [15] that have been thought to underlie the structural evolution into larger domains. Note that algorithms which flexibly allow for discontinuous fragments are making a resurgence, as in *Geometricus* [16], for analyzing structural embeddings.

In a recent study that developed a deep generative approach to protein structural relationships, using the Urfold model of protein structure in a framework called *DeepUrfold*, 20 superfamily-specific, sparse 3D-CNN variational autoencoders (VAEs) were trained for 20 different, highly-populated CATH superfamilies [17]. These DeepUrfold-trained models were shown to be agnostic to topology, as architecturally-similar SH3/OB proteins with artificially-constructed loop permutations yielded similar evidence lower bound–based (ELBO) scores; most significantly, applying community-detection methods (as stochastic block models) to the patterns of ELBO similarities led to the SH3 and OB domains clustering into similar groupings (with some intermixing). All of those findings were consistent with the prediction that the SH3 and OB comprise a distinct urfold (in this case, the SBB).

This paper explores—and seeks to begin *explaining*—the models from [17] in more depth, by applying an approach known as layer-wise relevance propagation (LRP). In principle, explainable AI techniques such as LRP can be used to understand which atoms in the input structure are 'important', based on their spatial locations and biophysical properties (and, really, any other sorts of features that one encodes in the model). In or our $\text{all}_{superfamilies} \times \text{all}_{domains}$ analysis, we look at the ELBO likelihood of a domain x under under *DeepUrfold* VAE models M_i and M_j for superfamilies i and j respectively. Functionally conserved regions from both M_i and M_j should positively affect the the likelihood under both models and therefore should have high LRP scores. Focusing on the two specific urfolds described above, i.e. the small β-barrels (SBB) and the phosphate-binding loop (PBL)–containing proteins, we show how LRP can be used to identify cross-model functionally important atoms; achieving that task, in turn, forms the foundation for identifying and characterizing new discontinuous fragments or ancestral peptides.

4.2 Results

4.2.1 Small β-barrels

We first investigated the SH3-specific (2.30.30.100), DeepUrfold-derived VAE model. This model was trained using all energy-minimized domain structures from the SH3 superfamily along with hand-crafted biophysical features, as described in [17]. We first attempted to subject representative SH3 domains through the SH3 model and calculated relevance scores during backpropagation. Promisingly, all of the residues that were previously identified in the SBB's "conserved hydrophobic core" [8] were labelled as relevant according to our LRP calculation (Fig. 4.2B). Next, we found that a specific, highly-conserved glycine in the second β-strand, known from years of manual analysis as being conformationally important in allowing the strand to bend [18], was deemed to be 'relevant' (Fig. 4.2C). Finally, we show that many contacts between strands $\beta4$ and $\beta5$, such as comprise the subunit interface in the 'Sm' variety of SH3-based oligomeric rings, are also labelled as important (Fig. 4.2D). From these initially promising results, we suspect that LRP can identify functionally important atoms, such as are learned as part of the latent space in the DeepUrfold-based superfamily models.

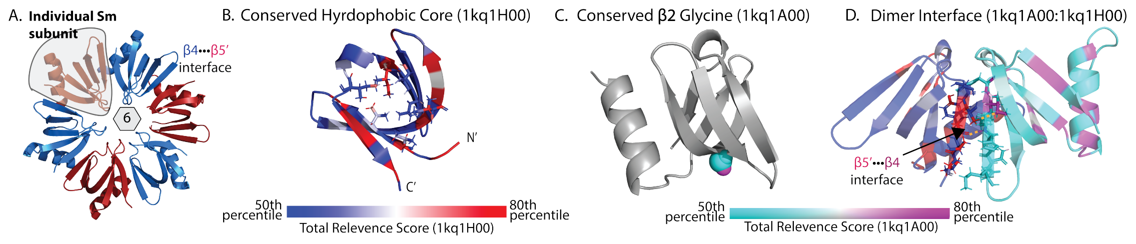

A. Individual Sm subunit
β4•••β5′ interface
6
B. Conserved Hyrdophobic Core (1kq1H00)
N′
C′
50th percentile
80th percentile
Total Relevence Score (1kq1H00)
C. Conserved β2 Glycine (1kq1A00)
D. Dimer Interface (1kq1A00:1kq1H00)
β5′•••β4 interface
50th percentile
80th percentile
Total Relevence Score (1kq1A00)

Figure 4.2: LRP Identifies Conserved and Structurally Important Regions in SH3 Domains. A) SH3 domains, and specifically those of the Sm/Sm-like proteins, tend to self-assemble into oligomeric rings of $n=5$, 6, or 7 SH3 domain subunits [18]. B) All SH3 domains, as well as other domains with the SBB urfold, have a conserved hydrophobic core [8]. LRP identifies residues in the core by being 80th percentile or greater, displayed using the spectrum from blue$\rightarrow$white$\rightarrow$red with range of 50th to 80th percentile all of the 1kq1H00 relevance scores. C) The phylogenetically conserved, structurally critical $\beta2$ glycine [18] is detected by LRP. The color scale for 1kq1A00 with spcetrum cyan$\rightarrow$white$\rightarrow$magenta with range 50th to 80th percentile of all of the 1kq1A00 relevance scores. D) By manual/visual inspection of the 1kq1H00:1kq1A00 dimer interface, we can see that important atoms from both 1kq1H00 roughly align with important atoms in 1kq1A00 (yellow dashed lines). 1kq1H00 is shown same color scale as in (B), and 1kq1A00 is shown with the same with the same color scale as in (C).

4.2.2 Phosphate-binding Loop (PBL)–Containing Proteins

We next tested the outcome of an $\text{all}_{superfamily} \times \text{all}_{domain}$ approach for the PBL urfold. That is, we trained two separate DeepUrfold VAE models, one for the canonical Rossmann Fold (3.40.50.720) and another for P-loop NTPases (3.40.50.300), and then we subjected representative domains from both superfamilies through both VAEs. Due to the importance of phosphate binding, we investigated residues that were known to specifically bind phosphates: the glycine-rich loop and the Walker B motif's aspartic acid on one of the edge strands of the PBL theme [12]. For all combinations of domain × model—i.e., (i) `Rossmann domain`→`Rosmann model`, (ii) `Rossmann domain`→`P-Loop NTPase model`, (iii) `P-Loop NTPase domain`→`Rossmann model`, and (iv) `P-Loop NTPase domain` → `P-Loop NTPase model`—we find that LRP correctly identifies the glycine-rich loop and Walker B Asp with relevance scores >=75th percentile, shown in Fig 4.3. Because the important atoms are predicted regardless of the DeepUrfold model, even for the model that is trained on domains annotated from a different CATH superfamily, we expect that these important residues, shared by both the Rossmann and PBL families, can be used to identify common fragments that comprise a joint Rossmann/PBL urfold.

4.3 Methods

4.3.1 DeepUrfold-Explain and VAE Model

In a recent paper that introduced DeepUrfold, the authors developed: (1) a preprocessed dataset based on CATH superfamlies that includes biophysical properties for each atom along with energy-minimized domain structures; and (2) superfamily-

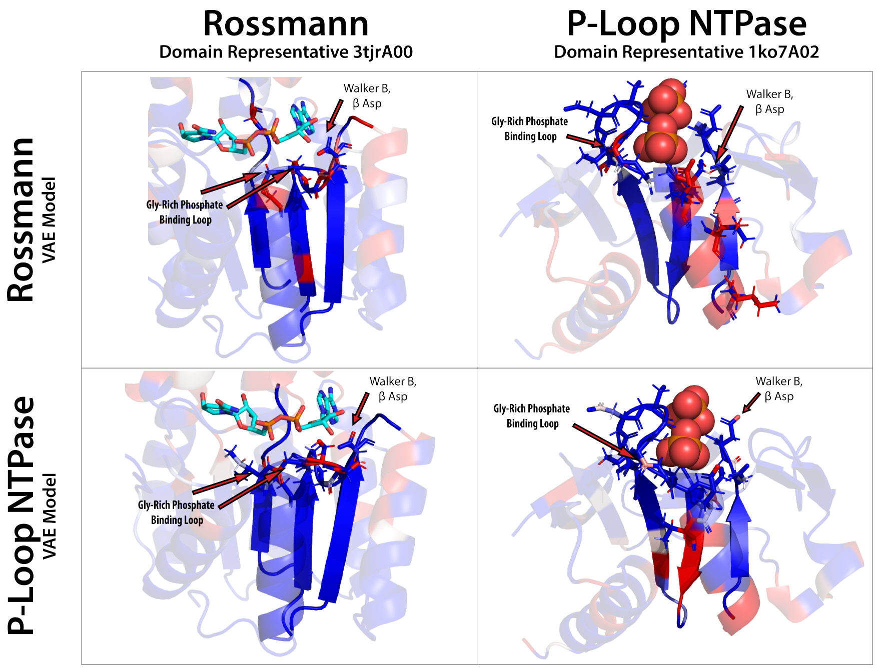


Figure 4.3: Important atoms in the Phosphate Binding Loop Urfold Identified via LRP. We subjected representative domains from the Rossmann (3tjrA00) and P-Loop NTPase (1ko7A02) superfamilies through each VAE model trained on all members of Rossmann *and* P-Loop NTPases respectively. Relevance scores are displayed on a spectrum from blue→white→red, using a range of 50th to 80th percentile of a given structure. Key atoms from residues that have been previously shown to be important in bridging these folds, namely the Walker B Asp motif and the glycine-rich loop [12], are selected by LRP having an relevance score >=75th percentile. Included ligands highlight phosphate-binding regions.

specific sparse 3D-CNN VAEs [17]. Energy-minimized domain structures from a single superfamily were voxelized using a kD-Tree to discretize atoms into 1^3 voxels in a 264^{33} volume and are rotated randomly by sampling the $SO(3)$ group to train a VAE model, modified from [19, 20], to create superfamily-specific models. Each voxel is annotated with biophysical properties of atoms that intersect it. Each VAE was trained using CATH's 30% sequence identity clusters, as defined by CATH for each superfamily, to create train (≈80%), test (≈10%), and validation (≈10%) splits. Hyperparameters used to construct the VAE were tuned using Weights & Biases [17].

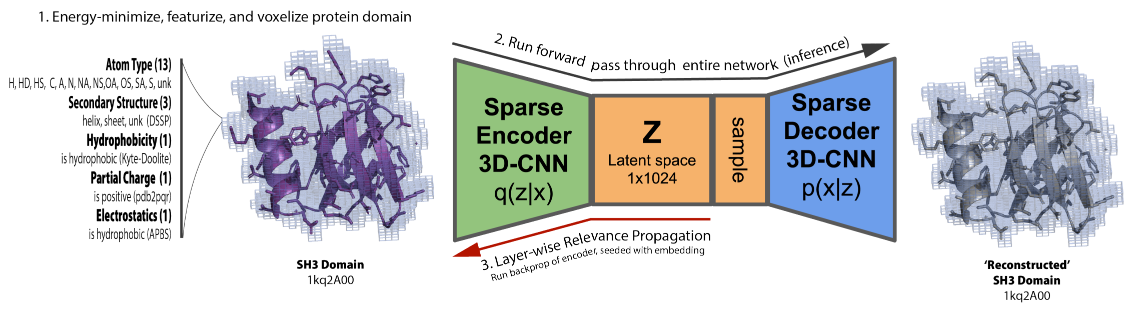

1. Energy-minimize, featurize, and voxelize protein domain
Atom Type (13)
H, HD, HS, C, A, N, NA, NS,OA, OS, SA, S, unk
Secondary Structure (3)
helix, sheet, unk (DSSP)
Hydrophobicity (1)
is hydrophobic (Kyte-Doolite)
Partial Charge (1)
is positive (pdb2pqr)
Electrostatics (1)
is hydrophobic (APBS)
SH3 Domain
1kq2A00
2. Run forward pass through entire network (inference)
Sparse Encoder 3D-CNN q(z|x)
Z
Latent space 1x1024
sample
Sparse Decoder 3D-CNN p(x|z)
3. Layer-wise Relevance Propagation
Run backprop of encoder, seeded with embedding
'Reconstructed' SH3 Domain
1kq2A00

Figure 4.4: DeepUrfold-Explain Identifies Important Atoms in Input Structures via LRP of Superfamily-specific VAEs. Relevant atoms are predicted for a given protein domain by: 1) obtaining an energy-minimized, featurized domain structure from a pre-calculated dataset and voxelized; 2) running the inference stage of pre-trained superfamily-specific VAE for the given domain; and finally, 3) running LRP during a backwards pass of only the encoder module, starting with the embedding of a given domain.

4.3.2 Layer-wise Relevance Propagation

Layer-wise relevance propagation defines a Relevance Score, as $R_j = \sum_k \frac{a_j \cdot \rho(\omega_{jk})}{\epsilon + \sum_{0,j} a_j \cdot \rho(\omega_{jk})} R_k$, where j is the current layer, a_j are the activations from the current layer, ρ is LRP rule, and ω_{jk} are the weights from the previous layer. LRP starts with the embedding of a given domain in a backwards pass. For layer j, a forward pass is run with the same data that was used as input into the layer j (the denominator) which is compared to the Relevance value from the previous layer, R_k. A backwards pass is then run using the data from the relevance weighted forward pass. Finally, the value from the backwards pass is compared to the output of the original input of the current layer [21].

We follow [22] and use rules LRP-0 for the lowest 60% of layers, ϵ-LRP (γ=0.25) for the middle 20%-60% of layers, and γ-LRP (γ=0.25) for the top 20% of layers. LRP-0 is the base rule where $\rho(x) = x$ and $\epsilon = 0$, which finds the contributions of each nueron to the final activation. ϵ-LRP is the base rule where $\rho(x) = x$, but $\epsilon > 0$, which helps remove noise and sparsify the explanations in terms of the input. γ-LRP sets $\rho(x) = x + \gamma x^{+}$, where x^{+} only includes positive relevance scores (all others set to 0) and $\epsilon = 0$, which is used to remove negative contributions.

Finally, we create a total relevance score by aggregating all relevance scores in a voxel by summing the relevance scores for every feature in that voxel and then we map voxels to atoms by taking the average total relevance score from all of the voxels that intersect a given atom based on a kD-Tree with radius the size of the atoms van der walls radius.

We adapted PyTorchLRP from [23] to add MinkowskiEngine layers [19] and regularization [24].

4.3.3 Cross-Model Fragment Identification

We subjected 2674 representative domains from 20 different superfamilies to 20 superfamily-specific VAEs, saving all LRP results. Residues containing any atom >= 75th percentile from a given structure were extracted to create a set of 53,480 (dis-)continuous fragments. For each community identified with Stochastic Block Modelling of the bipartite graph formed from the $\text{all}_{superfamilies} \times \text{all}_{domains}$ approach [17], we used foldseek [25] to cluster all LRP structures from domains present in each community that were processed through all superfamilies represented by the community (TM-Align global alignment). We select the LRP structure cluster representative from the most populated cluster in each community, resulting in the top 20 ‘potential urfolds.’

4.4 Conclusion

Machine learning for proteins is extremely difficult, partly due to the fact that all proteins are related via evolution [26]. It is important to know if a given model accurately represents reality or is giving garbage results. Explainable AI techniques and Layer-wise Relevance Propagation (LRP) alleviate these problems by allowing us to compare known biophysical properties of a given protein to a model’s prediction. We were able to show that LRP correctly selects structurally important and conserved atoms in SH3 domains, showing that the model is learning superfamily-specific features. Because the models are topologically-agnostic, we were also able to show that LRP can find important atoms from structures that exhibit ‘architectural similarity despite topological variability,’ specifically the phosphate binding loops in Rossmann and P-Loop NTPases. In the future, we plan to identify and verify more common fragments and ancestral peptides by aligning and clustering ‘important’ regions from the cross-model fragments while comparing them to known databases of potentially discontinuous fragment libraries, e.g. from shapemers [27], Fuzzle2.0 [28], ancestral peptides [14], themes [15], or TERMs [29].

Bibliography

[1] Philip E. Bourne, Eli J. Draizen, and Cameron Mura. The curse of the ribbon. *PLoS Biology*, Accepted 2022.

[2] Vamsi Nallapareddy, Nicola Bordin, Ian Sillitoe, Michael Heinzinger, Maria Littmann, Vaishali Waman, Neeladri Sen, Burkhard Rost, and Christine Orengo. CATHe: Detection of remote homologues for CATH superfamilies using embeddings from protein language models. *bioRxiv*, 2022.

[3] Michael Heinzinger, Maria Littmann, Ian Sillitoe, Nicola Bordin, Christine Orengo, and Burkhard Rost. Contrastive learning on protein embeddings enlightens midnight zone. *NAR Genomics and Bioinformatics*, 4(2), 06 2022. lqac043.

[4] Tymor Hamamsy, James T. Morton, Daniel Berenberg, Nicholas Carriero, Vladimir Gligorijevic, Robert Blackwell, Charlie E. M. Strauss, Julia Koehler Leman, Kyunghyun Cho, and Richard Bonneau. TM-Vec: Template modeling vectors for fast homology detection and alignment. *bioRxiv*, 2022.

[5] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 12(6):654–669.e3, June 2021.

[6] Ian Sillitoe, Nicola Bordin, Natalie Dawson, Vaishali P Waman, Paul Ashford, Harry M Scholes, Camilla S M Pang, Laurel Woodridge, Clemens Rauer, Neeladri Sen, Mahnaz Abbasian, Sean Le Cornu, Su Datt Lam, Karel Berka, Ivana Hutařová Varekova, Radka Svobodova, Jon Lees, and Christine A Orengo.

CATH: increased structural coverage of functional space. *Nucleic Acids Research*, 49(D1):D266–D273, November 2020.

[7] Cameron Mura, Stella Veretnik, and Philip E. Bourne. The *Urfold*: Structural similarity just above the superfold level? *Protein Science*, 28(12):2119–2126, November 2019.

[8] Philippe Youkharibache, Stella Veretnik, Qingliang Li, Kimberly A. Stanek, Cameron Mura, and Philip E. Bourne. The small β-barrel domain: A survey-based structural analysis. *Structure*, 27(1):6–26, January 2019.

[9] Claudia Alvarez-Carreño, Petar I Penev, Anton S Petrov, and Loren Dean Williams. Fold evolution before LUCA: Common ancestry of SH3 domains and OB domains. *Molecular Biology and Evolution*, 38(11):5134–5143, August 2021.

[10] Claudia Alvarez-Carreño, Rohan J Gupta, Anton S. Petrov, and Loren Dean Williams. The evolution of protein folds by creative destruction. *bioRxiv*, 2022.

[11] Vishal Agrawal and Radha KV Kishan. Functional evolution of two subtly different (similar) folds. *BMC Structural Biology*, 1(1):1–6, 2001.

[12] Liam M Longo, Jagoda Jabłońska, Pratik Vyas, Manil Kanade, Rachel Kolodny, Nir Ben-Tal, and Dan S Tawfik. On the emergence of P-Loop NTPase and Rossmann enzymes from a beta-alpha-beta ancestral fragment. *eLife*, 9, December 2020.

[13] Kirill E. Medvedev, Lisa N. Kinch, R. Dustin Schaeffer, and Nick V. Grishin. Functional analysis of Rossmann-like domains reveals convergent evolution of topology and reaction pathways. *PLOS Computational Biology*, 15(12):e1007569, December 2019.

[14] Vikram Alva, Johannes Söding, and Andrei N Lupas. A vocabulary of ancient peptides at the origin of folded proteins. *eLife*, 4, December 2015.

[15] Sergey Nepomnyachiy, Nir Ben-Tal, and Rachel Kolodny. Complex evolutionary footprints revealed in an analysis of reused protein segments of diverse lengths. *Proceedings of the National Academy of Sciences*, 114(44):11703–11708, October 2017.

[16] Janani Durairaj, Mehmet Akdel, Dick de Ridder, and Aalt D J van Dijk. Geometricus represents protein structures as shape-mers derived from moment invariants. *Bioinformatics*, 36(Supplement_2):i718–i725, December 2020.

[17] Eli J. Draizen, Stella Veretnik, Cameron Mura, and Philip E. Bourne. Deep generative models of protein structure uncover distant relationships across a continuous fold space, 2022.

[18] Cameron Mura, Peter S. Randolph, Jennifer Patterson, and Aaron E. Cozen. Archaeal and eukaryotic homologs of Hfq: A structural and evolutionary perspective on Sm function. *RNA Biology*, 10(4):636–651, April 2013.

[19] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal convnets: Minkowski convolutional neural networks. pages 3075–3084, 2019.

[20] JunYoung Gwak, Christopher B Choy, and Silvio Savarese. Generative sparse detection networks for 3D single-shot object detection. 2020.

[21] Alexander Binder, Grégoire Montavon, Sebastian Bach, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. 2016.

[22] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: An overview. pages 193–209, 2019.

[23] Moritz Böhle, Fabian Eitel, Martin Weygandt, and Kerstin Ritter. Layer-wise relevance propagation for explaining deep neural network decisions in MRI-based alzheimer's disease classification. *Frontiers in Aging Neuroscience*, 11, jul 2019.

[24] Erico Tjoa, Guo Heng, Lu Yuhao, and Cuntai Guan. Enhancing the extraction of interpretable information for ischemic stroke imaging from deep neural networks. 2019.

[25] Michel van Kempen, Stephanie S. Kim, Charlotte Tumescheit, Milot Mirdita, Johannes Söding, and Martin Steinegger. Foldseek: fast and accurate protein structure search. *bioRxiv*, 2022.

[26] Sean Whalen, Jacob Schreiber, William S. Noble, and Katherine S. Pollard. Navigating the pitfalls of applying machine learning in genomics. *Nature Reviews Genetics*, 23(3):169–181, November 2021.

[27] Janani Durairaj, Joana Pereira, Mehmet Akdel, and Torsten Schwede. What is hidden in the darkness? Characterization of AlphaFold structural space. *bioRxiv*, 2022.

[28] Noelia Ferruz, Florian Michel, Francisco Lobos, Steffen Schmidt, and Birte Höcker. Fuzzle 2.0: Ligand binding in natural protein building blocks. *Frontiers in Molecular Biosciences*, 8, August 2021.

[29] Craig O. Mackenzie, Jianfu Zhou, and Gevorg Grigoryan. Tertiary alphabet for the observable protein structural universe. *Proceedings of the National Academy of Sciences*, 113(47), November 2016.

[30] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv*, dec 2013.

4.5 Appendix

4.5.1 Variational Autoencoders

Each variational autoendcoder model learns a normal distribution for each superfamily using the 'reparameterization trick' to allow for backpropagation through random variables. This makes only the mean (μ) and variance (σ) differentiable, while sampling from the normally distributed random variable ($\mathcal{N}(0, \mathbf{I})$). That is, the latent variable posterior $\mathbf{z}$ is given by $\mathbf{z} = \mu + \sigma \bigodot \mathcal{N}(0, \mathbf{I})$, where $\bigodot$ denotes the Hadamard (element-wise) matrix product and $\mathcal{N}$ is the 'auxiliary noise' term ([30]).

Chapter 5

Epilogue

This dissertation has explored the possibility of creating and organizing a new, re-imagined protein universe, with a reproducible, systematic and yet flexible framework enabled by machine learning. Ideally, such a system would allow for pairwise relationships to be detected and allowed for (i.e., they would be *representable*) between proteins that are remotely homologous—even if the relationship is extremely faint and barely detectable. Perhaps such distantly-related proteins ought to be bridged together in a new *Urfold* model of protein interrelationships, which recognizes the existence of small peptide fragments that can be potentially discontinuous in sequence [1]? In terms of a hierarchical representation of the protein universe, the Urfold is conceived of as a new entity that sits between the 'architecture' and 'topology' strata (to use the terms from CATH), allowing for the phenomenon of "*architectural similarity despite topological variability*" [1]. In this scheme, the strict hierarchical nature of available databases, wherein a given protein is assigned to one mutually-exclusive bin or another, is no longer required—i.e., the protein universe makes more sense when viewed as a network instead of a tree, wherein, for example, an individual protein domain may well be thought of as belonging to more than one 'bin'. This new view of the protein universe can (i) give insights about how proteins could have evolved pre-LUCA, via small fragments; (ii) identify potential new protein-ligand and protein-protein interactions, based on known interactions in exceedingly remote homologs;

and (iii) aid in structure prediction and design, by providing a methodological foundation for creating less biased train & validation splits, with fewer remote homologs cross-polluting between the two datasets.

Available hierarchical representations of the protein universe, such as CATH [2], SCOP [3, 4], and ECOD [5], paved the way for biologists to think about the protein structure universe as clusters of evolutionarily, structurally and functionally related proteins. Such representations are powerful because they reduce the extremely high-dimensionality protein structure space into something a human brain can comprehend. This type of reductionism, often referred to as 'Wittgenstein's ladder' [6] or 'Lie-to-children' [7], is an important step in the process of human understanding, as it attempts to distill an intractably complicated subject/topic into a more comprehensible form via a simplified model. An analogous device in structural biology (at the level of individual proteins) is the cartoon ribbon diagrams, which we believe has engendered what we have called 'The Curse of the Ribbon': cartoon diagrams of protein 3D structures, which highlight the secondary structures, are themselves a reductionist viewpoint that ignores biophysical properties, protein dynamics, and other nuances, though such representations have allowed for countless insights into structural biology [8]. However, we are now in an era where newer tools can be developed to help address what we believe are the shortcomings of these classic representations, and get to the next rung of the ladder. In broad terms, the development of such tools is a central goal of this thesis.

The work described in this dissertation can be largely viewed as a form of *biomolecular data science*. What does that mean? Here, I will summarize and reflect on the content of this thesis by considering the work from the perspective of the Five Pillars of Data Science: (i) Data Acquisition; (ii) Data Integration & Engineering; (iii)

Machine Learning & Analytics; (iv) Visualization & Dissemination; and (v) Ethics, Law, Policy, & Social Implications [9]. Finally, I will conclude with thoughts on future work and directions, were the lines of work described in this thesis to be continued.

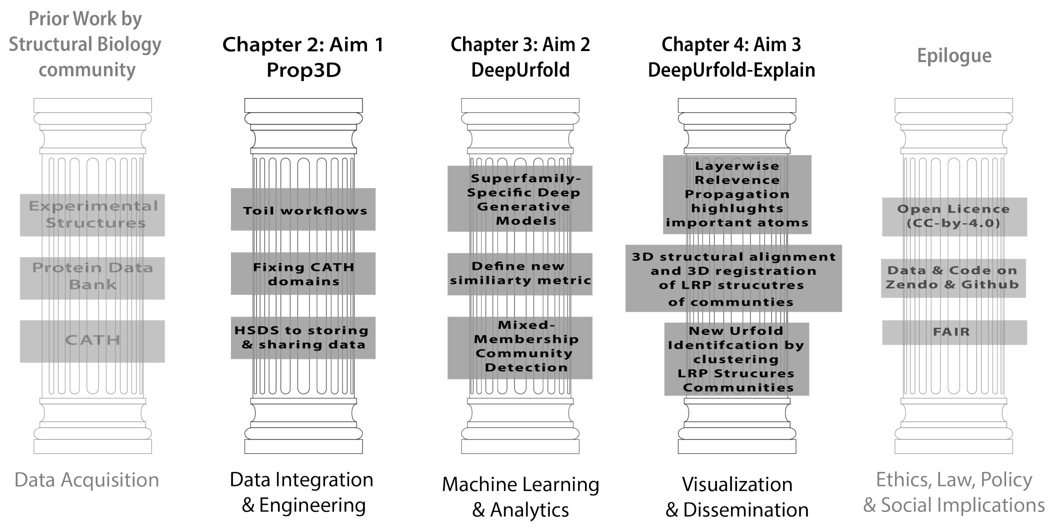

Prior Work by Structural Biology community
Chapter 2: Aim 1 Prop3D
Chapter 3: Aim 2 DeepUrfold
Chapter 4: Aim 3 DeepUrfold-Explain
Epilogue
Experimental Structures
Protein Data Bank
CATH
Toil workflows
Fixing CATH domains
HSDS to storing & sharing data
Superfamily-Specific Deep Generative Models
Define new similiarty metric
Mixed-Membership Community Detection
Layerwise Relevence Propagation highlughts important atoms
3D structural alignment and 3D registration of LRP strucutres of communties
New Urfold Identifcation by clustering LRP Strucures Communities
Open Licence (CC-by-4.0)
Data & Code on Zendo & Github
FAIR
Data Acquisition
Data Integration & Engineering
Machine Learning & Analytics
Visualization & Dissemination
Ethics, Law, Policy & Social Implications

Figure 5.1: This Thesis & the Five Pillars of Data Science. Here, we see that the contents of this thesis can be fairly cleanly mapped to the five pillars of data science; more details on the pillars can be found in [9].

5.1 Data Acquisition

The field of Structural Biology has had a rich history in open-access science and data sharing/availability. For example, the Protein Dank Bank has been collecting and sharing protein structures determined by X-ray Crystallogphy since 1971 — from ≈75 structures to a total of ≈200K that are available today that are freely accessible to everyone around the world [10]. Because of that history, I was able to freely obtain all of the information that served as the raw/primary data for this dissertation's projects—including all 3D structures of proteins that have been experimentally determined (deposited into the PDB), as well as extracted domain structures from CATH (a public database of domain interrelationships [2]). Therefore, the realm of Data Acquisition was itself not a Chapter or major area of activity in this thesis; as with all of scientific research, this dissertation's work built upon that of past researchers, in this case decades of structural biology efforts that have amassed nearly 200,000 structures in the PDB (and worked their way into downstream databases like CATH and SCOP).

5.2 Data Engineering $\leftrightarrow$ Chapter 2 (Prop3D)

Pursuing any domain-specific question, be it in structural biology or any other realm of basic science, demands that certain computational tools and utilities be in place; these resources may take the form of lightweight toolkits, software libraries, monolithic codebases, datasets, etc. Ideally, these tools will strike a balance between being (i) general, such that they can be used in different data science applications; and (ii) specific enough to address our given problem of identifying the urfold. My effort to explore the Urfold, via the DeepUrfold methodology (Chapter 3), required me to develop a useful set of tools and datasets that are both useful (for the calculation at hand) and extensible (for future calculations). From the perspective of data science, this machine learning codebase corresponds to the 'data engineering' pillar.

Specifically, I developed Prop3D, a resource to create and share biophysical atomic properties for all domains in CATH. This is a data engineering task because, as detailed in Chapter 2, I needed to automate the 'cleaning' of every domain structure by adding missing atoms and residues, protonating all atoms, and energy minimizing the structure in order create the most biophysically relevant protein structure. I processed the entire CATH dataset in a massively parallel workflow using Toil [11] and saved and shared the data using the Highly Scalable Data Service (HSDS) [12]. In pursuing that code-development work, I sought to strike a balance in writing code that was not overly general-purpose, that would be extensible (modular, amenable to being refactored and otherwise adapted), and that would provide the necessary functionality for (i) structural bioinformatics tasks as well as (ii) deployment on a massively-parallel scale.

5.3 Data Analytics ↔ Chapter 3 (DeepUrfold)

Next, I developed DeepUrfold, a deep generative model based on one-class Variational Autoencoders (VAEs), to learn about relationships between protein structures at the level of CATH homologous superfamilies. This undertaking essentially created a new distance measure between a given domain structure and a given CATH superfamily, quantified by the likelihood that the given structure came from a specific superfamily-model (versus others); this calculation was achieved via the Evidence Lower Bound (ELBO) metric, which is a concept from the underlying variational Bayesian theory of VAEs. To examine how variations in topology affect each VAE-based DeepUrfold model, I created fictitious folds(/topologies) which had the same architecture, yet systemically permuted loops to modify the topology for representative domains from the SH3 and OB superfamilies. Note that this exercise effectively serves as a negative control for the core Urfold concept of "architectural similarity despite topological variability". Each permuted or 'rewired' domain structure was found to yield similar ELBO scores to its wild-type (unscrambled) domain when run through its superfamily VAE model, thus showing that the VAEs were indeed agnostic of topology [13]; this result, in turn, serves as a necessary (albeit not sufficient) condition for being able to use the DeepUrfold machine learning framework to identify distinct urfolds.

As described in Chapter 3, I ran all representative domains from a single superfamily and subjected them to the VAE trained on all domains from the same superfamily, saving the embeddings for each domain; this process was repeated for all 20 highly-populated CATH superfamilies that were chosen for our initial study. We used the Uniform Manifold Approximation and Projection (UMAP) approach to dimensionality reduction in order to probe the high-dimensional latest spaces of the individual, superfamily-specific VAE DeepUrfold models. Doing so, we found that the embeddings for all domains, across all subfamilies, were organized fairly smoothly by their secondary structure content—all-α proteins were quite visually distinct from all-β, with α/β occurring intermediately. Most promising, these results are reassuringly consistent with earlier studies by others (e.g., Sung-Hou Kim, William Taylor, Rachel Kolodny) and, moreover, suggest that these global properties of protein fold space can be captured by the deep neural network architecture of DeepUrfold [13].

Finally, I performed an all-vs-all analysis—in this case, $\text{all}_{superfamilies} \times$ $\text{all}_{domains}$—by subjecting all representative domains from the 20 highly-populated superfamilies and all superfamily-specific VAEs, saving the ELBO score for each combination; note that this calculation yields what is essentially a fully-connected bipartite graph. To analyze this graph, I developed a new mixed-membership clustering method to find interrelationships between CATH domains using a Stochastic Block Model on the adjacency matrix of the graph, wherein the edges are weighted by ELBO scores. I show that there is a large degree of intermixing between 'true'/annotated' CATH superfamilies compared to the predicted SBM communities. The domains in an SBM community all have similar ELBO scores to every superfamily-specific VAE, suggesting they are more similar based on biophysical, physicochemical and evolutionary properties in combination with geometry, versus purely 3D structure/geometry [13]. It is also plausible that they share common structures of biophysical properties & geometry that could potentially be noncontiguous (i.e., structural fragments) in 3D space; this latter possibility, of sub-domain structural fragments, is a question explored in Chapter 4.

5.4 Visualization & Dissemination $\leftrightarrow$ Chapter 4 (DeepUrfold-explain)

Finally, I used an Explainable AI technique, termed Layer-wise Relevance Propagation (LRP), to understand *how* each model manages to create an embedding that successfully captures or 'represents' a given set of similar domains (i.e., the particular superfamily being considered). To be more explicit, in our context of protein structural biology the '***how***' in the last sentence really means the underlying physical/molecular basis for the machine learning results—in this case, what features might a trained VAE DeepUrfold model have learnt about one superfamily that distinguishes it from other superfamilies, in terms of its being an optimal match to a particular domain? In effect, the answer to this question can be viewed as providing a way to define superfamily $\mathcal{A}$ versus $\mathcal{B}$ versus $\mathcal{C}$ and so on—this, indeed, is central to DeepUrfold's view of protein groupings and interrelationships. Following the similar $\text{all}_{superfamilies} \times \text{all}_{domains}$ approach as described in Chapter 3, I added a step to calculate all relevant voxels and atoms in a backwards pass of the model. A most exciting initial result is that, at least for the proteins examined thus far, those atoms which are predicted to be important by LRP also have known biochemical and biophysical features that have flagged them as being significant (e.g., from past experimental biochemical or structural characterization in the literature).

Next, we attempt to identify common substructures among domains in the same SBM community mapped to (i) atom space; and (ii) voxel space. In both cases we perform an all-by-all 3D superposition of atom or voxels to create a distance metric, cluster on those distances, and find regions of all the cluster members that overlap with the cluster representative. These approaches yield discontinuous fragments of structure and is the first step at creating an automatic, rigorous, systemic, and reproducible definition of the Urfold.

5.5 Ethics

All of my datasets, code, and analyses comply with the 'FAIR' principles for research best practices, meaning they are Findable, Accessible, Interoperable, and Reusable. In particular, all the information in this dissertation has been made publicly available under a Creative Commons license (CC-by-4.0); furthermore, any new datasets that build from this work must follow these FAIR guidelines. In other words, the dataset must: (1) be easy to find, with appropriate metadata to facilitate searching; (2) be available to access all of the data easily; (3) be able to integrate with other data and software; (4) and be replicable, so that others can reproduce/adapt/etc. our workflows without undue effort. To be FAIR, all of this dissertation's code, datasets, and analyses have been released on Github and Zenodo, with sufficient documentation and descriptions on how to use them.

5.6 Future Directions

Given more time to work on this project, I would first apply DeepUrfold to every CATH superfamily to systematically and exhaustively predict Urfolds across the protein universe. I was able to show that new urfolds could be identified within those 20 CATH superfamilies that we considered with high population density; it would be extremely interesting to apply DeepUrfold to the remaining six thousands superfamilies, if computational resources and time were not a bottleneck. In terms of the 'community structure' and general patterns of interrelationships of protein superfamilies across all of fold space, it would also be interesting to explore in greater detail the mixed-membership aspect of the Stochastic Block Modelling in Chapter 3.

5.6.1 Method Development

Next, since the start of this thesis, several improved deep learning models for protein structure have been developed and appeared in the recent literature — indeed, deep learning is a fast-moving target, for applications in biology and beyond! It would be exciting to apply these very new approaches to learn more about the Urfold. Most notably, Equivariant Neural networks [14] would improve our current 3D-CNN approach because it would alleviate the computational overhead associated with rotating all 3D structures randomly (we only sample 30 random rotations, which is likely missing possible orientations), and certainly it would accelerate training of our models.

AlphaFold2 [15], along with other efforts, notably RoseTTAFold [16], have been the most significant achievements in deep learning & structural biology to predict protein structures. By combining a novel MSA transformer with Equivariant Neural

Networks, they were able to predict protein structures with significantly greater accuracy than any previous approach. These models have also been adapted to perform protein design [17]. If Explainable AI techniques were used to interrogate the AlphaFold2 modeling process, perhaps the results could be used to further refine the definition of an Urfold (and detect individual urfolds)?

Other deep neural network models that would aid in the detection of urfolds could be the recent large language models (LLM), graph neural nets, and geometric deep learning. LLMs have been recently shown to be useful in finding remote homologs [18], and in structure prediction & design [19, 20]. Graph-based neural nets are another important deep neural network to encode protein structure information [21, 22], and can even be combined with LLMs to obtain more sensitive results [23]. Finally, I would also like to use geometric deep learning models such as dMaSIF [24], for "Differentiable Molecular Surface Interaction Fingerprints", to explore the Urfold. Such might be possible because this method can compare similarities in surface features, not the protein core, so it can be considered to be agnostic of topology. In all of these cases, the resultant models can be used to learn about the Urfold, in molecular and structural biology terms, by creating deep generative networks that can be explored with LRP or other explainable AI techniques.

5.6.2 Application

Finally, longer-term I would like to use the Urfold model of protein structure in order to interrogate protein interaction networks. From what we know about SH3 and OB interactions, I believe it would be fruitful to discover and characterize more instances of this phenomenon.

When I first started my thesis work, my plan was to study proteins involved at the host-parasite interaction (HPI) interface, especially proteins from apicomplexan parasites (single-celled eukaryotic pathogens that cause malaria and toxoplasmosis). Many HPIs involve proteins with permuted structures, highlighting the potential importance of the Urfold model of protein structures and structural relationships (or at least an Urfold-like approach). Apicomplexan Urfolds can be seen in the 6-Cys surface proteins, which are hypothesized to have evolved from a mammalian ephrin protein, obtained in turn via horizontal gene transfer events [25]. Apicomplexans are thought to have co-opted an ephrin because the latter proteins serve as ligands that play key roles in cell:cell interactions, which the parasites can use to 'trick' the host into binding with them. Ephrins can be seen to contain an Urfold that is shared with Immunoglobulins: they both have a seven-stranded β-sandwich architecture; however, their topology has been permuted [26]. Another example of an Apicomplexan urfold can be seen in the group of protease inhibitors known as serpins, which also have structures that are permuted relative to their human homologs [27, 28]. Identifying urfolds in pathogenic proteins can help us learn about the evolution of the immune system and immune evasion, allowing for the development of new drugs and vaccines to treat these pathogens.

Bibliography

[1] Cameron Mura, Stella Veretnik, and Philip E Bourne. The urfold: Structural similarity just above the superfold level? *Protein Science*, 28(12):2119–2126, Nov 2019.

[2] Ian Sillitoe, Natalie Dawson, Tony E Lewis, Sayoni Das, Jonathan G Lees, Paul Ashford, Adeyelu Tolulope, Harry M Scholes, Ilya Senatorov, Andra Bujan, Fatima Ceballos Rodriguez-Conde, Benjamin Dowling, Janet Thornton, and Christine A Orengo. Cath: expanding the horizons of structure-based functional annotations for genome sequences. *Nucleic Acids Research*, 47(D1):D280–D284, Jan 2019.

[3] Naomi K Fox, Steven E Brenner, and John-Marc Chandonia. Scope: Structural classification of proteins–extended, integrating scop and astral data and classification of new structures. *Nucleic Acids Research*, 42(Database issue):D304–9, Jan 2014.

[4] Antonina Andreeva, Dave Howorth, Cyrus Chothia, Eugene Kulesha, and Alexey G Murzin. Scop2 prototype: a new approach to protein structure mining. *Nucleic Acids Research*, 42(Database issue):D310–4, Jan 2014.

[5] Hua Cheng, R Dustin Schaeffer, Yuxing Liao, Lisa N Kinch, Jimin Pei, Shuoyong Shi, Bong-Hyun Kim, and Nick V Grishin. Ecod: an evolutionary classification of protein domains. *PLoS Computational Biology*, 10(12):e1003926, Dec 2014.

[6] Alastair Hannay, editor. *Cambridge texts in the history of philosophy: Kierkegaard: Concluding unscientific postscript.* Cambridge University Press, Cambridge, England, May 2009.

[7] Jack S Cohen and Ian Stewart. *Collapse of chaos.* Viking, London, England, June 1994.

[8] Philip E. Bourne, Eli J. Draizen, and Cameron Mura. The curse of the ribbon. *PLoS Biology*, In Review 2022.

[9] Cameron Mura, Eli J Draizen, and Philip E Bourne. Structural biology meets data science: does anything change? *Current Opinion in Structural Biology*, 52:95–102, Oct 2018.

[10] Stephen K. Burley, Helen M. Berman, Jose M. Duarte, Zukang Feng, Justin W. Flatt, Brian P. Hudson, Robert Lowe, Ezra Peisach, Dennis W. Piehl, Yana Rose, Andrej Sali, Monica Sekharan, Chenghua Shao, Brinda Vallat, Maria Voigt, John D. Westbrook, Jasmine Y. Young, and Christine Zardecki. Protein data bank: A comprehensive review of 3d structure holdings and worldwide utilization by researchers, educators, and students. *Biomolecules*, 12(10):1425, October 2022.

[11] John Vivian, Arjun Arkal Rao, Frank Austin Nothaft, Christopher Ketchum, Joel Armstrong, Adam Novak, Jacob Pfeil, Jake Narkizian, Alden D Deran, Audrey Musselman-Brown, Hannes Schmidt, Peter Amstutz, Brian Craft, Mary Goldman, Kate Rosenbloom, Melissa Cline, Brian O'Connor, Megan Hanna, Chet Birger, W James Kent, David A Patterson, Anthony D Joseph, Jingchun Zhu, Sasha Zaranek, Gad Getz, David Haussler, and Benedict Paten. Toil en-

ables reproducible, open source, big biomedical data analyses. *Nat. Biotechnol.*, 35(4):314–316, April 2017.

[12] Shweta Gopaulakrishnan, Samuela Pollack, B J Stubbs, Hervé Pagès, John Readey, Sean Davis, Levi Waldron, Martin Morgan, and Vincent Carey. restfulSE: A semantically rich interface for cloud-scale genomics with bioconductor. *F1000Res.*, 8:21, January 2019.

[13] Eli J. Draizen, Stella Veretnik, Cameron Mura, and Philip E. Bourne. Deep generative models of protein structure uncover distant relationships across a continuous fold space. *bioRxiv*, 2022.

[14] Fabian B. Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. SE(3)-transformers: 3D roto-translation equivariant attention networks. *arXiv*, 2020.

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

[16] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R Glassman, Andy DeGiovanni, Jose H Pereira, Andria V Rodrigues, Alberdina A van Dijk, Ana C

Ebrecht, Diederik J Opperman, Theo Sagmeister, Christoph Buhlheller, Tea Pavkov-Keller, Manoj K Rathinaswamy, Udit Dalwadi, Calvin K Yip, John E Burke, K Christopher Garcia, Nick V Grishin, Paul D Adams, Randy J Read, and David Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, August 2021.

[17] Christoffer Norn, Basile I. M. Wicky, David Juergens, Sirui Liu, David Kim, Brian Koepnick, Ivan Anishchenko, Foldit Players, David Baker, and Sergey Ovchinnikov. Protein sequence design by explicit energy landscape optimization. *bioRxiv*, 2020.

[18] Tymor Hamamsy, James T. Morton, Daniel Berenberg, Nicholas Carriero, Vladimir Gligorijevic, Robert Blackwell, Charlie E. M. Strauss, Julia Koehler Leman, Kyunghyun Cho, and Richard Bonneau. Tm-vec: template modeling vectors for fast homology detection and alignment. *bioRxiv*, 2022.

[19] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, and Alexander Rives. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022.

[20] Ruidong Wu, Fan Ding, Rui Wang, Rui Shen, Xiwen Zhang, Shitong Luo, Chenpeng Su, Zuofan Wu, Qi Xie, Bonnie Berger, Jianzhu Ma, and Jian Peng. High-resolution de novo structure prediction from primary sequence. *bioRxiv*, 2022.

[21] J Dauparas, I Anishchenko, N Bennett, H Bai, R J Ragotte, L F Milles, B I M Wicky, A Courbet, R J de Haas, N Bethel, P J Y Leung, T F Huddy, S Pellock, D Tischer, F Chan, B Koepnick, H Nguyen, A Kang, B Sankaran, A K Bera,

N P King, and D Baker. Robust deep learning-based protein sequence design using ProteinMPNN. *Science*, 378(6615):49–56, October 2022.

[22] Zuobai Zhang, Minghao Xu, Arian Jamasb, Vijil Chenthamarakshan, Aurelie Lozano, Payel Das, and Jian Tang. Protein representation learning by geometric structure pretraining. *"arXiv"*, 2022.

[23] Can Chen, Jingbo Zhou, Fan Wang, Xue Liu, and Dejing Dou. Structure-aware protein self-supervised learning, 2022.

[24] Freyr Sverrisson, Jean Feydy, Bruno E. Correia, and Michael M. Bronstein. Fast end-to-end learning on protein surfaces. *bioRxiv*, 2020.

[25] Silvia A Arredondo, Mengli Cai, Yuki Takayama, Nicholas J MacDonald, D Eric Anderson, L Aravind, G Marius Clore, and Louis H Miller. Structure of the plasmodium 6-cysteine s48/45 domain. *Proc. Natl. Acad. Sci. U. S. A.*, 109(17):6692–6697, April 2012.

[26] Julien Grassot, Manolo Gouy, Guy Perrière, and Guy Mouchiroud. Origin and molecular evolution of receptor tyrosine kinases with immunoglobulin-like domains. *Mol. Biol. Evol.*, 23(6):1232–1241, June 2006.

[27] Viviana Pszenny, Paul H Davis, Xing W Zhou, Christopher A Hunter, Vern B Carruthers, and David S Roos. Targeted disruption of toxoplasma gondii serine protease inhibitor 1 increases bradyzoite cyst formation in vitro and parasite tissue burden in mice. *Infect. Immun.*, 80(3):1156–1165, March 2012.

[28] S Ye, A L Cech, R Belmares, R C Bergstrom, Y Tong, D R Corey, M R Kanost, and E J Goldsmith. The structure of a michaelis serpin-protease complex. *Nat. Struct. Biol.*, 8(11):979–983, November 2001.