FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of master's thesis

Title:	Machine learning for the design of protein–protein interactions
Student:	Bc. Anton Bushuiev
Supervisor:	Dr. Ing. Josef Šivic
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2023/2024

Instructions

Protein–protein interactions are essential for biological processes. The development and possible treatment of diseases, such as cancer or stroke, are directly linked to specific properties of involved protein–protein interactions (Leader et al., 2008; Nikitin et al., 2022). Therefore, the design of proteins with desired binding properties is a central challenge for pharmacology. While several attempts at tackling this problem using machine learning have been made, the domain is relatively new, and there is still no reliable tool for engineering protein–protein interfaces. The thesis aims to approach this challenge using modern machine learning (Bronstein et al., 2021). More specifically, the objectives of the project are to:

1. Review state-of-the-art machine-learning methods for the design of protein interactions. Identify their limitations and benefits.

2. Explore the possibilities for addressing the identified limitations and improving the state-of-the-art methods for designing protein interactions. For example, a promising direction is self-supervised geometric deep learning from unlabeled crystallized protein–protein interactions to learn a new powerful neural base representation for protein interaction tasks.

3. Apply the selected representative tools and (optionally) the proposed new model to staphylokinase, a promising thrombolytic drug candidate.

Electronically approved by Ing. Magda Friedjungová, Ph.D. on 3 February 2023 in Prague.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Machine learning for the design of protein–protein interactions

Anton Bushuiev

Department of Applied Mathematics

Supervisors: Dr. Josef Šivic, Dr. Stanislav Mazurenko, Dr. Jiří Sedlář

May 4, 2023

Acknowledgements

First of all, I would like to express my lifelong gratitude to my parents for making my education possible and for all their support.

I am deeply grateful to Dr. Josef Šivic for providing me with the unique opportunity to work on this exciting project and for being an outstanding scientific advisor. I am sincerely thankful to Dr. Stanislav Mazurenko for his excellent co-supervision and the unique multidisciplinary expertise he has shared with me. I would also like to express my deep appreciation to Dr. Jiří Sedlář and Petr Kouba for their help and intellectual contribution to the development of the project.

Lastly, I want to convey my utmost appreciation to Prof. Jiří Damborský, MUDr. Jan Mičan, and Dr. David Bednář for all the insightful discussions on the problematics of protein design, which have been strongly enhancing my understanding of the domain.

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140) and also by the project National Institute for Neurological Research (Programme EXCELES, ID Project No. LX22NPO5107) - Funded by the European Union – Next Generation EU.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on May 4, 2023 . . . . . .

Czech Technical University in Prague

Faculty of Information Technology

Citation of this thesis

Bushuiev, Anton. *Machine learning for the design of protein–protein interactions*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2023.

Abstrakt

Cévní mozková příhoda patří celosvětově mezi hlavní příčiny úmrtí a invalidity, a mezi ostatními zdravotními poruchami představuje jednu z nejtěžších socioekonomických zátěží. V této práci aplikujeme nejmodernější metody strojového učení s cílem návrhu nové generace trombolytika stafylokinázy. Naše případová studie zdůrazňuje silné a slabé stránky existujících metod pro návrh interakcí mezi proteiny, které dále adresujeme vývojem nového modelu samořízeného geometrického hlubokého učení PPIFORMER. Předběžná analýza našeho přístupu ukazuje na jeho vysoký potenciál překonat omezení současných metod pro návrh protein–protein interakcí a stát se tak nástrojem nové generace pro návrh trombolytik a jiných léků.

Klíčová slova protein–protein interakce, proteinové inženýrství, stafylokináza, samořízené učení, geometrické hluboké učení

Abstract

Stroke is a leading cause of death and disability worldwide, resulting in one of the heaviest socioeconomic burdens of any disease kind. In this thesis, we apply state-of-the-art machine-learning methods with the goal of designing a next-generation thrombolytic staphylokinase. Our case study highlights the strengths and weaknesses of existing methods for the design of protein–protein interactions, which we further address by developing a novel self-supervised geometric deep-learning model PPIFORMER. The preliminary analysis of our approach demonstrates its high potential to overcome the limitations of current methods for designing protein–protein interactions and thus become a next-generation tool for the design of thrombolytics and other medicines.

Keywords protein–protein interactions, protein design, staphylokinase, self-supervised learning, geometric deep learning

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Stroke is a leading cause of death and disability worldwide, as well as one of the most frequent causes of dementia and epilepsy. The rapidly growing burden of stroke (102% increase in prevalent strokes and 143% increase in disability from 1990 to 2019) necessitates urgent measures. However, the high cost of well-established thrombolytics limits their widespread application, leading to the bulk of the global burden (86% of deaths and 89% of disability) residing in lower-income and lower-middle-income countries (Feigin et al., 2022). The staphylokinase (SAK) protein offers hope for overcoming the hard burden of stroke. This therapeutic protein has the potential to be a widely affordable, as well as safer, alternative to the best existing thrombolytics (Nikitin et al., 2022).

The primary bottleneck limiting the widespread clinical use of staphylokinase is its weak tendency to interact with plasmin, a protein present in blood. Together, these two proteins effectively catalyze the cleavage of blood clots and restore blood circulation. Therefore, in order to make the staphylokinase mechanism efficient, one needs to redesign a part of the protein for higher affinity towards plasmin. This can be achieved by introducing several favorable mutations, accurately selected from billions of possible and prevalently-disruptive ones. The complex combinatorial nature of protein design substantially exceeds human capabilities and motivates the application of machine learning.

In our work, we apply the best available machine-learning models to propose a set of favorable staphylokinase mutations. At the time of writing, the selected variants are being experimentally validated at Loschmidt Laboratories (Masaryk Uni-

versity, Brno). From a broader perspective, our case study reveals that although reliable machine-learning models exist for general-purpose protein design, there is a significant gap in models developed specifically for the task of protein–protein interaction design. This gap is critical, as protein–protein interactions are involved in nearly all cellular processes (Alberts et al., 2015). We reason that the primary cause of the unreliability of the existing methods is their dependence on small annotated data. Consequently, we develop a self-supervised training scheme and a geometric deep learning model to train on a thoroughly pre-processed dataset of potentially all known structures of protein–protein interactions. Our preliminary analysis of the model demonstrates the concept of the proposed approach. In summary, the main contributions of this thesis are the following:

1. We employ the most-advanced machine-learning models available to select a limited number of promising staphylokinase mutations for experimental validation at Loschmidt Laboratories. To achieve the robust selection, we develop a consensus algorithm that accounts for optimizing multiple protein properties while utilizing the collective knowledge of diverse predictive models.

2. We analyze and prepare existing protein–protein interaction datasets, revealing the severe limitations of their conventional usage. To perform the large-scale analysis, we develop a fast algorithm for comparing protein–protein interfaces.

3. We develop PPIFORMER, a self-supervised geometric deep-learning model that overcomes the data scarcity limitation of existing machine-learning models for the design of protein–protein interactions. Preliminary analysis of our approach indicates its strong potential.

Structure and notation

In Chapter 2 of the thesis, we cover the necessary biochemistry and machine learning background. In Chapter 3, we discuss the state of the art in machine learning relevant to the design of protein–protein interactions. Chapter 4 is dedicated to our case study of staphylokinase design. We describe our application of existing machine-learning methods to redesign the staphylokinase protein for higher affinity towards plasmin. In Chapter 5, we describe the analysis and preparation of

big protein–protein interaction data. We highlight the severe limitations of their standard usage and propose measures for their effective utilization. Finally, in Chapter 6, we present the PPIFORMER model trained on the prepared data. We provide the proof of concept for our approach by demonstrating its promising preliminary capabilities.

Throughout the thesis, we use standard mathematical notation. Linear algebra objects are marked in bold: uppercase letters for matrices (e.g. $\mathbf{M} \in \mathbb{R}^{r,c}$) and lowercase letters for vectors (e.g. $\mathbf{v} \in \mathbb{R}^{d}$). Then, the i-th row and the i,j-th element of a matrix $\mathbf{M}$ are denoted as $\mathbf{m}_i$ and $m_{i,j}$, respectively, while the i-th element of a vector $\mathbf{v}$ is written as v_i.

CHAPTER 2

Background

In this chapter, we present the essential concepts and terminology required to understand the thesis. First, we introduce the biochemical background, followed by a brief description of the main ideas in modern deep learning. For a more in-depth understanding of biochemistry, we recommend the book by Alberts et al. (2015), while for further insight into deep learning, we recommend the books by Goodfellow et al. (2016) and Bronstein et al. (2021).

2.1 Biochemistry

2.1.1 Proteins

Proteins are the key building blocks of cells, accounting for the majority of a cell's dry mass. These macromolecules typically contain thousands of atoms and are critical to nearly every cellular function, including catalyzing chemical reactions and acting as antibodies, transporters, or hormones. Proteins also perform specialized roles as antifreeze molecules, elastic fibers, or luminescence generators. Moreover, proteins transport organelles within the cytoplasm and facilitate communication between cells. Remarkably, the countless functions of proteins stem from relatively simple combinatorial principles underlying their structure. In fact, a unique combination of small building blocks determines the specific function of each protein, enabling it to bind and process other molecules or carry out a variety of other specialized tasks. Overall, proteins play an essential role in cellular biology and are critical to the proper functioning of living organisms. Understanding the principles underlying protein structure and function is critical to advancing

our knowledge of fundamental biological processes and developing new therapies for diseases (Alberts et al., 2015).

Proteins, like all macromolecules in a cell, are composed of a specific sequence of subunits that define their **primary structure** (Figure 2.1 A, left). These subunits are known as **amino acids**, and there are 20 different types found in most living organisms. All amino acids have a common structure, consisting of an **alpha-carbon** atom (C_α), **amino group** and **carboxyl group**, and a distinctive **side chain** (Figure 2.1 B). The side chains vary in chemical composition, giving each amino acid unique properties. Roughly half of the amino acids are **polar** (or **hydrophilic**), meaning they form hydrogen bonds with water molecules, while the other half are **nonpolar** (or **hydrophobic**), tending to cluster together in water solution. The nonpolar amino acids include alanine (Ala, A), glycine (Gly, G), valine (Val, V), leucine (Leu, L), isoleucine (Ile, I), proline (Pro, P), phenylalanine (Phe, F), methionine (Met, M), tryptophan (Trp, W), and cysteine (Cys, C). Among the polar amino acids, aspartic acid (Asp, D) and glutamic acid (Glu, E) are negatively charged, while arginine (Arg, R), lysine (Lys, K), and histidine (His, H) are positively charged. The remaining five polar amino acids are neutrally charged, with some of their fragments being positive and others negative, thereby compensating each other. These five amino acids are asparagine (Asn, N), glutamine (Gln, Q), serine (Ser, S), threonine (Thr, T), and tyrosine (Tyr, Y).

The complementarity of the carboxyl and amino groups enables amino acids to connect into **chains**. This is done by ribosomes (mixture of RNAs and proteins), which tightly link amino acids (or **residues**) one by one with **covalent** (strong, electron-sharing) bonds to form protein chains. While the whole chain (or **sequence**) is being constructed, the protein **folds** to adopt a specific three-dimensional shape. This is achieved by the formation of a complex network of **non-covalent** (weaker, no electron sharing) bonds. These include mostly **hydrogen bonds**, which occur between atoms that strongly sacrifice their electrons in other, covalent bonds and atoms that, conversely, pull electrons, resulting in positive and negative charges respectively. During the first 5 milliseconds of folding, hydrogen bonds between amino hydrogens and carboxyl oxygens shape the protein's **secondary structure** (Figure 2.1 A, middle), thus defining the high-level geometry of the protein **backbone** (the composition of all non-side chain atoms). Then, for up to a second, the protein structure is being refined by the formation of

other bonds between the amino acids until the molecule achieves its final, energetically minimal state. The coordinates of atoms in this state define the **tertiary structure** of the protein (Figure 2.1 A, right). The huge number of possible tertiary structures of proteins leads to the impressive variety of their functions.

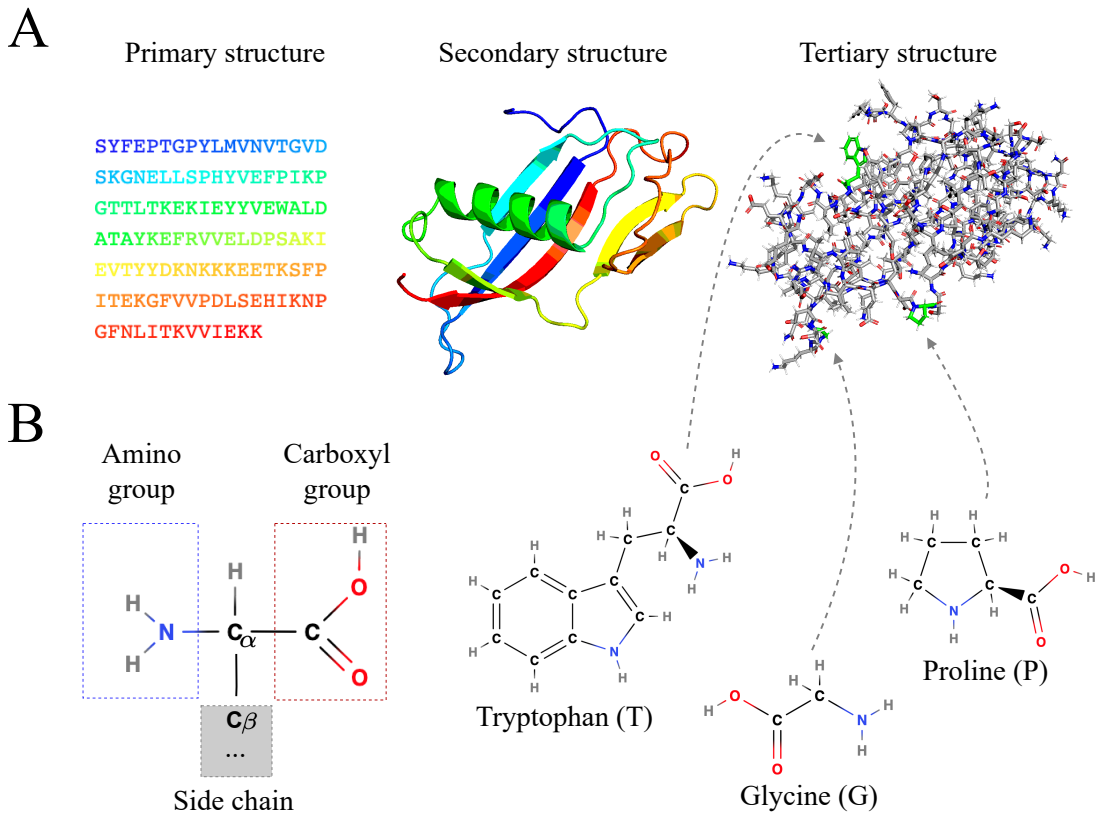

Figure 2.1: Protein structure. A) Three levels of protein structure. Primary structure is a linear sequence made of 20 possible amino acids. The protein chain is colored in sequential order. Secondary structure captures common folding patterns such as α-helices (visualized as helices), β-sheets (arrows) and loops (the rest). Tertiary structure defines the three-dimensional arrangement of atoms. The visualized protein is staphylokinase (PDB code `2SAK`). **B)** Generic structure of an amino acid (left) and three selected amino acids (right). While tryptophan is the heaviest and bulkiest of all twenty amino acids, glycine does not contain a side chain at all, and the side chain of proline is covalently linked to its amino group.

2.1.2 Protein–protein interactions

Proteins rarely function alone, with protein-protein interactions (PPIs) playing a critical role in various biological processes, such as cell signaling, metabolism, and gene regulation (Alberts et al., 2015). These interactions occur when two or more proteins **bind** to form a **complex** and can be transient or long-lasting. Understanding PPIs is crucial for drug design, as disruptions in PPIs can cause diseases such as

cancer and neurodegenerative disorders (Hardcastle, 2017). For instance, uncontrolled protein aggregation underlies Creutzfeldt–Jakob and Alzheimer's diseases (Marques et al., 2023). Conversely, PPIs between antibodies and antigens are favorable since they serve to identify and block foreign objects. Thus, drugs can be designed to either enhance or inhibit the interaction, modulating the biological process dependent on it. The study of PPIs is an ever-expanding field, with researchers continually developing new techniques to comprehend these intricate interactions and design more effective therapeutics.

The structure of a protein complex is known as a **quaternary structure**. When the complex involves multiple protein chains, it is often called a protein **oligomer** or, in the case of two chains, a **dimer**. Unlike **multi-domain proteins** (i.e. proteins of a single chain but several parts that fold independently and may have different functions), protein complexes involve separate chains, each having its own backbone. Nevertheless, the principles that govern PPIs are similar to those guiding protein folding. The formation of a quaternary structure is governed by the establishment of many weaker, non-covalent attractions between two protein surfaces. These attractions include previously mentioned hydrogen bonds, **ionic interactions** (i.e. attractions between complementarily charged amino acids), **Van der Waals forces** (i.e. weak bondings between atoms due to their fluctuating electrical charges), or **hydrophobic bonds** (i.e. non-polar amino acids getting close to each other to "avoid" interacting with water). In this way, PPIs are highly-specific, depending on the strong complementarity of the corresponding surface shapes and charge distributions.

The **binding affinity** of a protein–protein interaction (i.e. the "willingness" of proteins to interact) is typically measured by the free-energy change upon binding ΔG, defined as

$$\Delta G = G_{free} - G_{complex}, \tag{2.1}$$

where G_{free} and $G_{complex}$ are the free energies of the systems of unbound and bound proteins, respectively. Free energy G can be understood as an amount of useful energy that can be harnessed to do work, or a thermodynamic potential, a reduction in energy that is necessary for a transition, e.g. from free to complex, to be spontaneous under the given conditions. This means that only bindings with a negative ΔG are energetically favorable and can occur spontaneously. This also implies that the lower the ΔG value is, the "easier" it is for the binding to

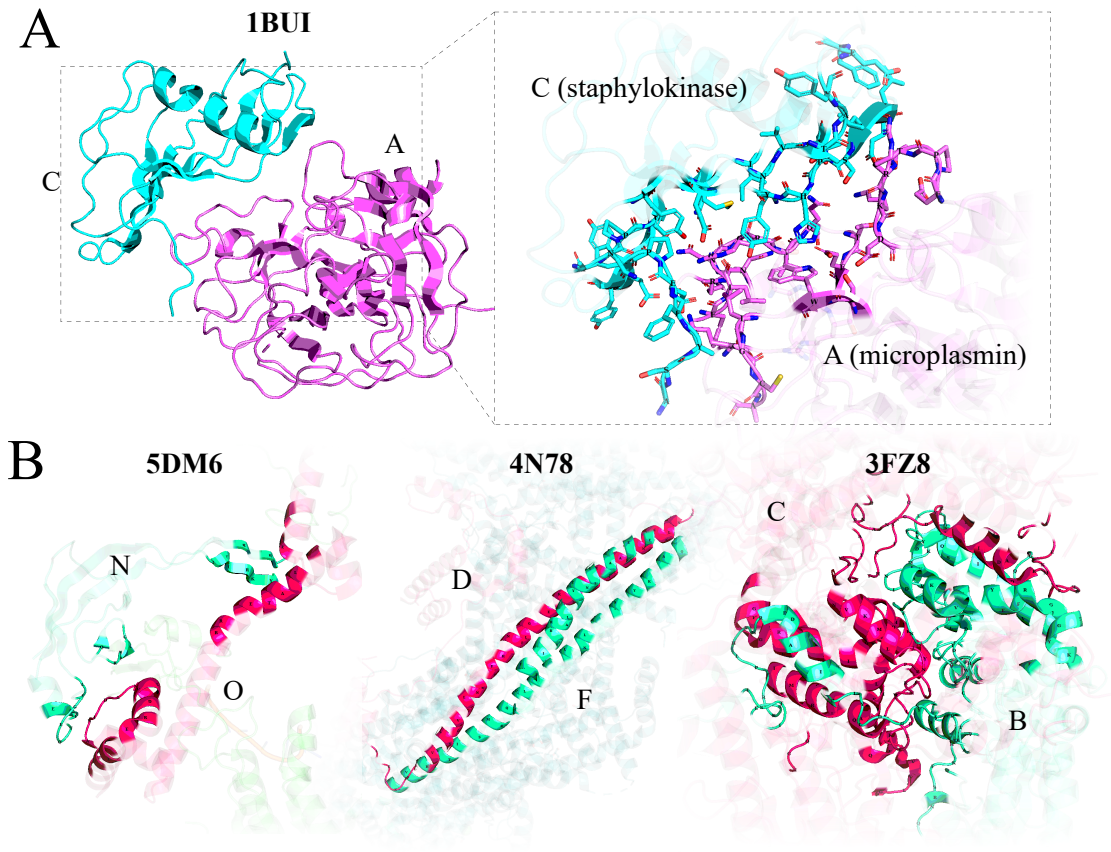

Figure 2.2: Examples of protein–protein interactions. A) Staphylokinase-microplasmin complex (left) and its interface (right). **B)** The diversity of protein–protein interfaces. **Left)** Disjoint interface. **Middle)** Extensive helical binding. **Right)** Extremely intertwined interaction. The highlighted interfaces are based on 6Å distance between heavy atoms.

occur. Alternatively, ΔG can be expressed as the logarithmic ratio between the dissociation and association rates

$$\Delta G = RT \ln K = RT \ln \frac{k_{off}}{k_{on}}, \tag{2.2}$$

where R and T are positive gas and temperature environmental constants, and k_{off} and k_{on} are the rates of PPI unbinding and binding, respectively. The ratio K is commonly referred to as the equilibrium constant. If the proteins tend to bind (i.e. $k_{on} > k_{off}$), the ΔG is negative, corresponding to an energetically favorable reaction. From this point of view, a lower ΔG corresponds to a higher fraction of interacting proteins in equilibrium. Additionally, Equation 2.4 outlines a standard way to measure the free energy change experimentally (Alberts et al., 2015).

When the quaternary structure of a protein–protein interaction is available, the most common approach to study the PPI is by examining its **interface**, that

is, the residues directly involved in the interaction. There is no standard definition for a protein–protein interface, but most studies define interfacial residues as those that are in close proximity to the partner. For instance, Gao and Skolnick (2010b) define an interface as a set of residues with at least one heavy (i.e. non-hydrogen) atom no more than 4.5Å[1] away from a heavy atom in the other protein. Table 2.1 provides other examples of interface definitions. An important property of protein–protein interfaces is their **buried surface area (BSA)**, which is defined for a dimer with two chains A and B as

$$BSA = ASA^{A}_{free} + ASA^{B}_{free} - ASA_{complex}, \tag{2.3}$$

where ASA represents the available (i.e. exposed to the environment) surface area of a corresponding structure measured in Å^2. BSA is a significant determinant of binding affinity and is sometimes used to define an interface (Kastritis and Bonvin, 2013; Levy, 2010). The interfacial residues that contribute to the establishment of an interaction the most are known as **hotspots** or **hot regions** (Keskin et al., 2008).

Considered atoms	Maximum distance	Reference
Heavy atoms	4.5Å	Gao and Skolnick (2010b)
Heavy surface atoms	5Å	Shin et al. (2023)
All atoms	5Å	Mirabello and Wallner (2018)
Heavy atoms	6Å	Townshend et al. (2019)
C_α atoms	8Å	Ganea et al. (2021)
C_β atoms	10Å	Watson et al. (2022)
Heavy atoms	10Å	Jankauskaitė et al. (2019)

Table 2.1: Examples of protein–protein interface definitions.

2.1.3 Protein design

Protein design, also known as protein engineering, is a cutting-edge technique that enables the creation of proteins with enhanced or novel functional properties. Modern experimental biochemistry allows modifying the genetic information of a cell to produce virtually any desired protein sequence. As such, protein design

[1] Ångström (Å) is a metric unit of length. 1Å = 10^{-10}m.

aims to identify advantageous mutations (i.e. protein sequence substitutions that increase a property of interest) in natural (**wild-type**) proteins or to create entirely new proteins through a process called *de novo* protein design.

A primary challenge in protein engineering is the combinatorial complexity of the protein space. An average protein of 400 residues can have $19 \times 400 = 7600$ **single-point mutations** (i.e. substitutions of a single residue), and the number of, for example, three-point mutations counts up to tens of billions. The search for beneficial mutations has a "needle in a haystack principle" as most mutations have unfavorable effects, while in practice one is interested in higher-order, **multi-point mutants** with ten or more substitutions to achieve a significant impact (Laroche et al., 2000). Navigating through this vast space of potential mutants is complicated by the phenomenon of **epistasis**, which refers to the non-additive effects of mutations. Miton and Tokuriki (2016) analyzed nine case studies and found that half of the effects of multi-point mutations are unpredictable from single-point mutation data. For instance, combining two highly favorable single-residue substitutions could result in a disruptive joint effect.

One of the directions of protein design is the design of protein–protein interactions. For example, one may be interested in redesigning the interface of an antibody to enhance its binding affinity towards the antigen. In such cases, the effects of mutations can be measured using the $\Delta\Delta G$ metric, defined as

$$\Delta\Delta G = \Delta G_{mut} - \Delta G_{wt}, \tag{2.4}$$

where ΔG_{mut} and ΔG_{wt} correspond to the binding affinity of mutated and wild-type complexes, respectively. Negative $\Delta\Delta G$ values indicate favorable mutations that increase affinity, while positive values signify a disruptive effect on binding.

Traditional protein-design approaches utilize evolutionary statistics or physics-based simulations to determine the favorability of mutations. For instance, a position-specific scoring matrix (PSSM) enables the estimation of evolutionary plausibility for all single-point substitutions (Beckstette et al., 2006). To construct the matrix, a set of sequences homologous (i.e. evolutionarily related, structurally similar) to the studied one is required. Then, one can align sequences and count the probabilities of amino acids at each position. The scores derived from the probabilities can help to narrow down the range of possible mutations to those that have been naturally selected during evolution. While this approach can ensure

safer mutation selection, it may also lead the design away from highly-favorable novel substitutions. Conversely, force field-based physics simulators such as FoldX and Rosetta provide estimates of ΔG and $\Delta\Delta G$ based on protein tertiary or quaternary structure, without relying on the evolutionary information (Schymkowitz et al., 2005; Das and Baker, 2008).

Some notable examples of hybrid methods combining evolutionary and physics-based calculations are HotSpot Wizard and Affilib. HotSpot Wizard integrates various traditional protein-design approaches, such as PSSM, FoldX and Rosetta, into a single software pipeline for comprehensive analysis of protein mutations. This method facilitates the selection of the most crucial residues for design (Sumbalova et al., 2018). Affilib is another software that can be employed to design protein–protein interactions with enhanced properties (Netzer et al., 2018). This method initially preselects a range of single-point substitutions based on PSSM scores and single-point $\Delta\Delta G$ estimates by Rosetta. It then uses Rosetta to exhaustively score the selected multi-point mutants.

2.1.4 Staphylokinase

An important case study for the design of protein–protein interactions are thrombolytics. Those are proteins that break up clots by activating fibrinolysis and converting the plasminogen protein to plasmin. The latter then degrades fibrin clots in blood, prompting the use of thrombolytics for the emergency treatment of an ischemic stroke, a heart attack, or a massive pulmonary embolism. The staphylokinase (SAK) protein is an attractive thrombolytic drug candidate. In comparison to the most commonly used alteplase, it is a smaller, more affordable, and highly specific agent. As a result, it has the potential to be a cost-effective and safer alternative for stroke treatment. Staphylokinase has already demonstrated its beneficial properties in multiple clinical trials (Nikitin et al., 2022). However, the primary limitation hindering its widespread clinical use is its low efficiency. Since the activity of SAK is directly related to its interactions with other proteins, our study aims to design the staphylokinase interface to improve its binding properties.

Figure 2.3 offers a more comprehensive understanding of the thrombolytic activity of staphylokinase. Upon introduction to the bloodstream, staphylokinase forms a complex with the human protein plasmin (or its truncated version microplasmin). In close proximity to fibrin clots, they jointly catalyze the generation

of additional plasmin molecules by plasminogen. The abundance of plasmins then effectively breaks down the clot, reopening the blood vessel and reestablishing blood circulation. The primary bottleneck of the mechanism is the low affinity of SAK towards plasmin, which is the main motivation of the thesis. Our objective is to redesign the interface of staphylokinase for increased affinity to plasmin. Furthermore, we take into consideration the hypothesis that SAK's activity may be constrained by its dimerization, and thus aim to reduce the potential for SAK–SAK interactions.

Introducing mutations into the interface of SAK, it is essential to preserve the protein's vital properties. Specifically, maintaining high stability is crucial, allowing the protein to retain its fold and continue functioning despite environmental fluctuations. Moreover, staphylokinase must be well-tolerated by the human body, meaning it should not be targeted by antibodies as a foreign object. Further, we refer to these two properties as simply **stability** and **immunogenicity**. The affinity of SAK for plasmin is simplified to **affinity**, while the tendency for SAK–SAK interactions is denoted as **dimerization**.

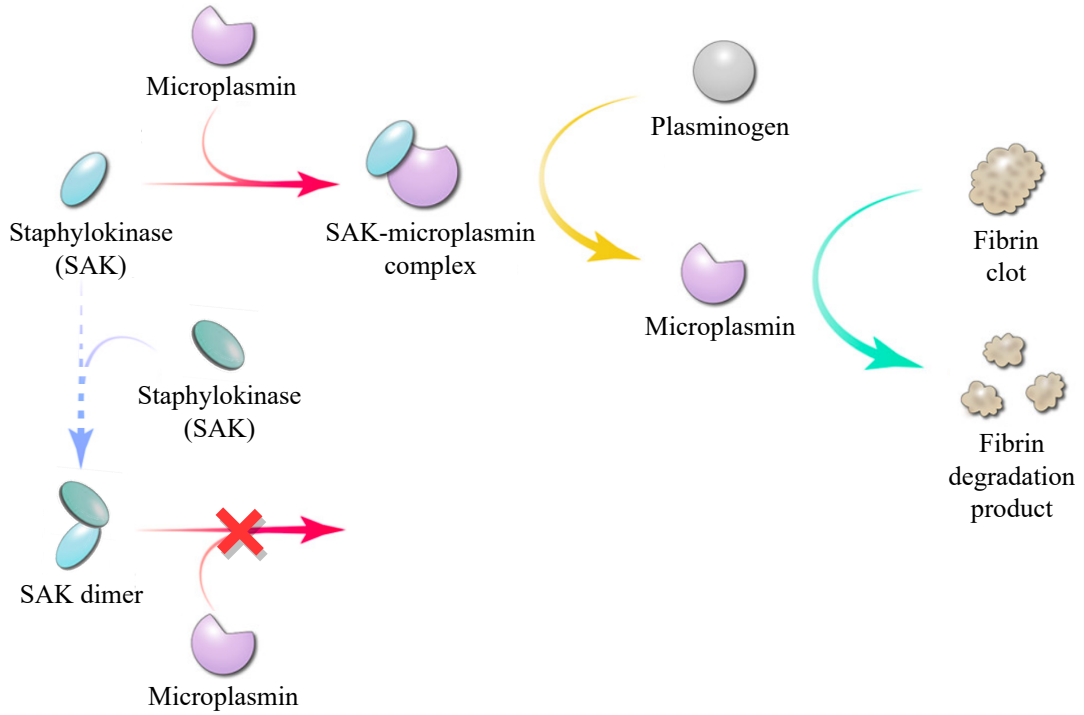

Figure 2.3: Thrombolytic mechanism of staphylokinase. The diagram is partially reproduced from Toul et al. (2022) and highlights protein–protein interactions relevant to the present study.

2.2 Machine learning

2.2.1 Deep learning

In general terms, machine learning is a method to utilize data to improve computer performance on specific tasks. There are four primary components in machine learning: a parametrized computer program f, commonly known as a **model**, a task(s) T along with the corresponding performance measure(s) P, and a dataset D (Goodfellow et al., 2016). For instance, a researcher in protein design may define a task T to associate a protein and its mutations with the $\Delta\Delta G$ value. In this case, the data D could be provided by experimental observations, and the performance measure P might be set as the absolute difference between a computer-generated $\Delta\Delta G$ value and the actual one. Then, the goal is to **train** the machine by implementing an algorithm that finds the best shape of f from a defined set $\mathcal{F}$:

$$\arg\max_{f \in \mathcal{F}} \sum_{(x,y) \in D} P(f(x), y), \tag{2.5}$$

where x and y represent the input and desired output in the data (e.g., mutated protein and $\Delta\Delta G$). Machine learning research primarily focuses on the development of more effective models f, performance measures P, tasks T, and datasets D, along with improved methods for optimizing Expression 2.5.

Deep learning is a subfield of machine learning which studies artificial neural networks. An artificial neural network (also known as a multi-layer perceptron, MLP) is a machine-learning model of the form

$$f = f_l \circ f_{l-1} \circ \cdots \circ f_1, \tag{2.6}$$

$$f_i(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{W}_i \mathbf{x} + \mathbf{b}_i), \tag{2.7}$$

where $\mathbf{x}$ is an input vector, $\mathbf{W}_i$, $\mathbf{b}_i$ are the matrix and vector parameters of the function that are being optimized (i.e. that define a set $\mathcal{F}$), and $\boldsymbol{\sigma}$ is an element-wise **non-linearity**, typically $\mathrm{ReLU}(x) = \max(0, x)$. Functions f_i are called **layers** and parameters $\mathbf{W}_i$ and $\mathbf{b}_i$ are typically referred to as **weights** and **biases**. In plain words, each layer of an artificial neural network applies a linear transformation (i.e. scaling, rotation or reflection) to a vector, then shifts it and applies a simple non-linear transformation.

Once a neural network is trained (i.e. the function $f \in \mathcal{F}$ with the best parameters $\mathbf{W}_i$, $\mathbf{b}_i$ according to Expression 2.5 is found), it is typically evaluated on

an independent set of data. In practice, one typically has a single dataset and, therefore, requires to split it into the **training** and **test folds** to evaluate the **generalization capacity** of the model by training it on one part and testing on the other. For example, the dataset of handwritten digits for image classification can be split by the people who author the writing (LeCun, 1998). Such an approach ensures that the evaluation of the model's performance corresponds to its practical deployment: to classify the writing of new, previously unseen, people. In many cases, there is no natural scheme to establish a data split, which poses a challenge for fair evaluation.

2.2.2 Geometric deep learning

A multi-layer perceptron can approximate practically any vector function (Lu et al., 2017; Hornik, 1991). Nevertheless, deep learning is known for its breakthroughs in computer vision, natural language processing and other domains where data extends beyond simple tabular representations. The success of deep learning can be largely attributed to the invention of ways to properly adapt neural networks to complex data such as images or sequences of words. For example, convolutional neural networks (CNNs) combine multi-layer perceptrons, so that they can efficiently operate on grids of pixels with RGB values. Essentially, a convolutional neural network iteratively applies a certain type of multi-layer perceptron to each local patch of an image, mapping pixels to higher-dimensional internal representations. By functioning locally, CNNs exhibit translation equivariance. In simple terms, this means that CNNs are insensitive to image translations, enabling data-efficient training.

Formally, the property of **equivariance** is defined with respect to a set of transformations[2] G. A function f is said to be G-equivariant if it satisfies

$$f(g(x)) = g(f(x)) \quad \text{for any} \quad g \in G \quad \text{and input} \quad x. \tag{2.8}$$

The translational equivariance of a convolutional neural network, therefore, means that translating an image x leads to the same translation of the network's output $f(x)$. For example, if f is designed to detect a cat in the photo, outputting a segmentation map (i.e. a photo colored to highlight the detected cat), the property

[2]Formally, G is called a symmetry group and must satisfy several natural properties. Additionally, we further simplify Equation (2.8) by having the same g acting on both sides of the equation, while formally it may have different representations.

of equivariance ensures that translating the input photo will result in the same translation of the colored output.

In many cases, one is, however, interested in the special case of equivariance known as **invariance**:

$$f(g(x)) = f(x) \text{ for any } g \in G \text{ and input } x, \tag{2.9}$$

which ensures that the output of a network remains entirely unaffected by the considered transformations. For example, invariance is desired when the objective is to classify whether an image contains a cat rather than to detect its location. Provably, an invariant function can be obtained by stacking several equivariant functions followed by an invariant one (Bronstein et al., 2021). Consequently, in practice, one typically builds invariant deep-learning models by stacking several equivariant functions followed by a simple invariant one. Referring back to the example of image classification, a common approach would be to employ an equivariant CNN, followed by a simple averaging of the final per-pixel representations and applying an ordinary classification multi-layer perceptron.

The architecture of virtually any existing neural network for complex data can be justified by the equivariance to a certain group of transformations (Bronstein et al., 2021). For example, state-of-the-art graph neural networks are equivariant to permutations of node neighborhoods, which is a central property of a graph. Similarly, for example, modern deep-learning models for learning from spherical data (e.g. to predict temperature on the globe) are constructed to be equivariant to spherical rotations. When learning from 3-dimensional objects such as protein structures one is particularly interested in being agnostic to the arbitrariness of the underlying coordinate system, and, therefore, in **SE(3)-equivariance**. SE(3) denotes a special Euclidean group in three dimensions, which represents the set of all 3-dimensional rigid-body transformations, i.e. combinations of translations and rotations. The principle of SE(3)-equivariance is illustrated in Figure 2.4 A.

2.2.3 Self-supervised learning

Another revolutionary paradigm in deep learning is self-supervision. The concept of self-supervised training involves the construction of the x, y training pairs artificially, from unannotated input x alone. By learning to solve a synthetic task, the model can acquire a general understanding of the input domain, which can enable

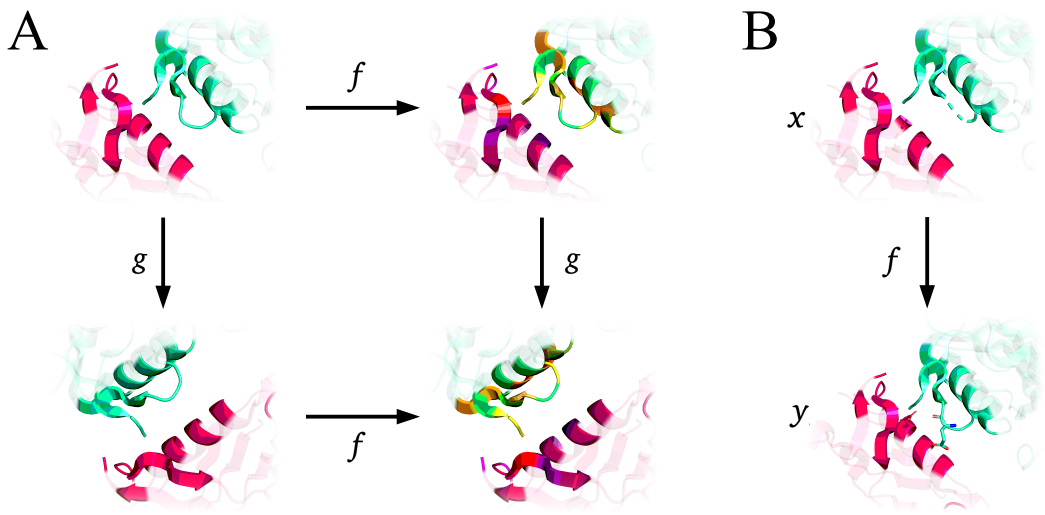

Figure 2.4: Principles of modern deep learning illustrated on a protein–protein interaction. A) The principle of equivarance, a central idea of geometric deep learning. The visualized property of SE(3)-equivariance guarantees that the deep-learning model f is insensitive to any rigid roto-translation of the input g, enabling data-efficient training. **B)** Self-supervised learning, a major deep-learning approach to overcome data scarcity. The figure visualizes an artificial task of completing a corrupted protein–protein interface. The illustrated interaction is the SAK–SAK dimer (PDB code `1C78`).

its easy adaption to many downstream problems of interest. Such an approach enables to overcome the limitations of traditional supervised learning, which often requires expensive and time-consuming labeling efforts. For example, in the absence of a sufficiently large annotated dataset for specific image classification, one can initially **pre-train** a model to fill in missing image parts. In this manner, a network can learn common patterns in image data and then be efficiently **fine-tuned** (i.e. adapted through further training) for classification. Figure 2.4 B illustrates a possible adaptation of the concept to the protein domain.

Successful application of self-supervised learning has led to the development of **foundation models**, large deep-learning models trained on vast quantities of unlabeled data via self-supervision. These models exhibit a general "understanding" of the domain and can be easily fine-tuned for various downstream tasks. Notable examples of foundation models include ChatGPT and GPT-4 (Bubeck et al., 2023). Most likely, the GPT models, or Generative Pre-trained Transformers, were mainly trained on the task of predicting artificially-masked subsequent text.

CHAPTER 3

Related work

This chapter is dedicated to the overview of the current advancements in machine learning for protein science, focusing on the design of protein–protein interactions. We do not aim to provide a comprehensive review of the approaches but rather highlight the most relevant methods along with the most noteworthy ones in the field.

3.1 Machine learning for proteins

In this section, we briefly review some of the most prominent achievements of deep learning on proteins. We first discuss the applications of deep learning to isolated proteins and then proceed with applications related to the interactions of proteins with other molecules in a living cell.

3.1.1 Single proteins

Arguably, the most outstanding application of deep learning in protein science is the development of AlphaFold2 (Jumper et al., 2021). This method solved the protein folding problem by demonstrating the ability to predict the three-dimensional structure of a protein from its sequence with a high accuracy, a challenge that had remained open for 50 years. At its core, AlphaFold2 relies primarily on an SE(3)-equivariant Transformer-like architecture, which operates jointly on the input sequence along with evolutionary-related ones to iteratively refine the positions and orientations of residues in the predicted structure. Protein folding remains an active research area, with new approaches proposing, for example, ways to re-

move the dependency on evolutionary information (Wu et al., 2022b) or enable the prediction of quaternary structures of protein complexes (Evans et al., 2021). Additionally, novel methods explore alternative architectures for protein folding, such as those based on diffusion generative modeling (Wu et al., 2022a) or large language models (Lin et al., 2022).

Many other successful applications of deep learning to protein-related problems draw inspiration from traditional deep learning domains. For instance, the primary structure of proteins has been actively studied through the lens of natural language processing. Methods such as ESM-2 and Ankh are Transformer-based language models capable of predicting secondary structure, fold type, solubility, or fluorescence of proteins solely from sequences of letters representing the amino acids (Lin et al., 2022; Elnaggar et al., 2023). Likewise, tertiary structures of proteins have been extensively analyzed through 3D convolutional neural networks. These applications include the prediction of protein interactions with water (Park and Seok, 2022) or with small molecules, often drugs (Li et al., 2019).

Similarly, graph neural networks have been playing an important role in tackling protein problems with deep learning (Zhou et al., 2020b). The ProteinMPNN model is a highly-prominent example of a graph neural network-based approach to learning from protein structures (Dauparas et al., 2022). ProteinMPNN has solved the problem known as inverse folding. In this task, a network is provided with a protein backbone of interest and predicts a sequence that can fold into the shape of the backbone. Internally, ProteinMPNN constructs a sequence in an autoregressive manner, amino acid by amino acid. Since the model implicitly predicts probabilities for each of the 20 possible amino acids at every step, it can also be employed to estimate the likelihood of specific substitutions for protein-design purposes.

3.1.2 Protein interactions

Recently, the tasks related to understanding how proteins interact with other molecules have become an active machine-learning research area. Tasks in this category include the prediction of protein–protein and protein–ligand docking. Docking is a molecular modeling task that aims to predict the mutual position and orientation of two molecules forming a complex. In the case of protein–protein docking, a network is given the structures of two proteins and learns to predict

the rototranslation that puts two proteins together in their native mode of interaction. Similarly, in a protein–ligand scenario, the task is defined to predict the position and orientation of a ligand (i.e. a small molecule, typically a drug) to describe where it binds to the protein.

EquiDock and EquiBind are the first attempts to tackle the docking problems with deep learning (Ganea et al., 2021; Stärk et al., 2022). The methods were shown to achieve performance competitive with traditional algorithms while being substantially faster. Essentially, these models rely on a graph neural network to find a match between graph representations of two molecules. Once the match is established they apply an alignment algorithm to estimate the rotation and translation that docks the molecules. Recently, diffusion generative models DiffDock and DiffDock-PP were shown to outperform the matching-based methods by learning to directly generate optimal transformations (Corso et al., 2022; Ketata et al., 2023).

Other tasks related to protein interactions include the closely related problems of docking pose scoring and binding energy prediction (Shen et al., 2020). Recently, Jin et al. (2023) introduced NERE, a deep-learning model that predicts the binding energy of protein–protein interactions using unsupervised deep learning. The primary concept behind the method involves maximizing the likelihood of native crystal structures of complexes. This is inspired by the fact that crystallized structures represent the lowest energy states.

Furthermore, machine learning can be employed to analyze protein–protein interactions at a more abstract level. For instance, given a network with nodes representing individual proteins and edges corresponding to various interaction types, one can apply machine learning to uncover previously unknown interactions or deduce their properties (Hu et al., 2020). Recently, Gao et al. (2023) proposed the HIGH-PPI graph neural network, which employs both a high-level network representation and a detailed residue-level graph representation of proteins to predict protein–protein interactions.

3.2 Machine learning for protein design

In recent years, machine learning has been increasingly utilized in protein design tasks. In this section, we begin by discussing machine learning techniques

for predicting the effects of mutations. These methods facilitate the screening of numerous protein mutations to identify the ones with the highest potential to improve a specific protein property. As the primary focus of our work is the design of protein–protein interactions, we provide a more detailed review of existing approaches for predicting $\Delta\Delta G$ upon binding. Additionally, we briefly mention generative techniques in protein design, which enable the creation of entirely new proteins that meet desired constraints.

3.2.1 Mutation effect prediction

Deep mutational scanning datasets offer millions of sequences annotated with the effects of introduced mutations (Fowler and Fields, 2014). Recent deep learning applications have explored the potential of predicting labels in deep mutational scanning data without relying on supervised training. For instance, DeepSequence utilizes a latent variable model to estimate the likelihood of mutated sequences (Riesselman et al., 2018). The more recent ESM-1v model leverages masking-based self-supervised pre-training of a large Transformer model on millions of unannotated sequences. Trained to predict missing amino acids in protein chains, the model was shown to be effective in scoring mutations. To estimate the score, ESM-1v first infers the probabilities of individual substitutions for a mutated position of interest and then calculates log-odds ratios that capture the relative plausibility of the wild-type and mutated sequences.

A similar approach was employed by Shroff et al. (2020) to score mutations based on self-supervised training from protein crystal structures. The proposed MutCompute utilizes a three-dimensional convolutional neural network trained to predict missing residues in atomic structures of proteins. Similar to the ESM-1v Transformer, MutCompute has proven to be practically useful for protein design. To select promising residue substitutions, one can mask its atoms in a protein structure and infer the probabilities of individual substitutions using the method. Residues with low predicted probabilities for wild-type amino acids can then be considered promising candidates for mutagenesis.

3.2.2 Mutation effect prediction for protein–protein interactions

While the described machine-learning methods for general-purpose protein design leverage self-supervision, current approaches for binding $\Delta\Delta G$ prediction heavily

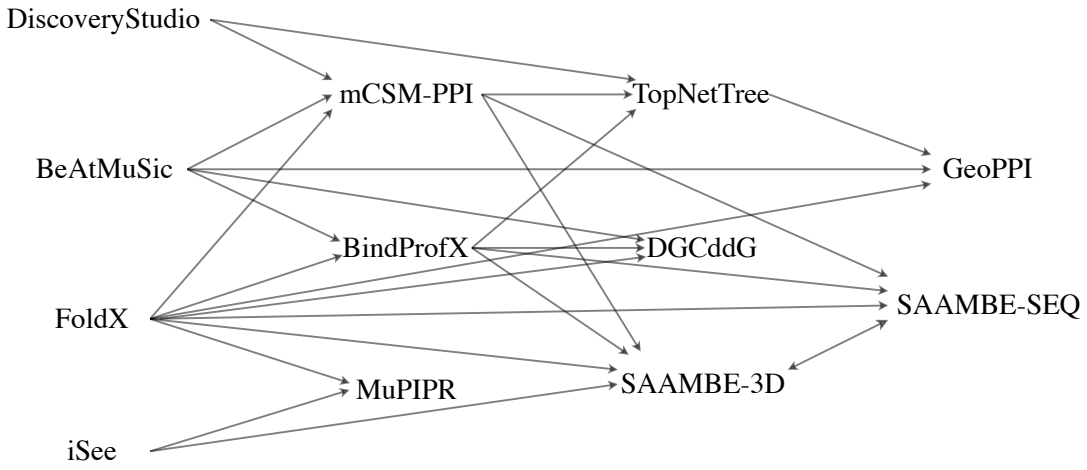

Figure 3.1: Evolution of $\Delta\Delta G$ predictors. The arrows illustrate published outperformance of methods on different subsets of the SKEMPI2 dataset. While the figure does not provide a comprehensive overview, it highlights key representative methods in the field.

rely on supervised training from the SKEMPI2 dataset. SKEMPI2 contains $\Delta\Delta G$-labeled mutations introduced to protein interfaces (see to Section 4.1.2 for a more detailed description). Earlier non-machine-learning approaches for predicting the effects of interfacial mutations rely on physics-based simulations or statistical potentials. For example, FoldX and DiscoveryStudio estimate $\Delta\Delta G$ by simulating a mutant structure with a force field (Delgado et al., 2019; Biovia et al., 2017), while BeAtMuSiC uses statistical potentials derived from coarse-grained protein models (Dehouck et al., 2013). BindProfX improves upon force-field simulations by incorporating evolutionary information (Xiong et al., 2017).

The iSee model represents one of the first approaches for predicting $\Delta\Delta G$ using machine learning on an earlier version of the SKEMPI2 dataset (Geng et al., 2019a). This method employs 31 structural, evolutionary, and energy-based features to estimate mutation effects with a random forest. Similarly, mCSM-PPI uses graph-based geometric features of the interface along with biochemical and evolutionary properties to train extra trees for $\Delta\Delta G$ prediction (Rodrigues et al., 2021). In the same line, SAAMBDE-3D employs 33 knowledge-based features representing the physical environment surrounding a mutation site (Pahari et al., 2020). TopNetTree enhances protein feature representation by utilizing extended persistent homology barcodes to capture topological and chemical features of the interface, which are then used to train gradient boosting trees (Wang et al., 2020). DGCddG does not rely on manually-crafted features but rather extracts them by

leveraging a graph neural network (Jiang et al., 2023). SAAMBE-SEQ focuses solely on sequences of interacting proteins but is limited to scoring single-point mutations (Li et al., 2021). MuPIPR, another sequence-based approach, leverages bi-directional recurrent neural networks to predict the mutation effects (Zhou et al., 2020a).

To the best of our knowledge, GeoPPI is the most advanced $\Delta\Delta G$ predictor (Liu et al., 2021). During the first step of inference, it constructs a mutated structure with FoldX. Then, it applies an attention-based graph neural network, pre-trained on a synthetic task of restoring correct rotations of residue side chains, to both wild-type and mutant-type structures to extract interface features. These features then serve as input for gradient boosting trees that estimate the mutation effect. The evolution of the described $\Delta\Delta G$ predictors is summarized in Figure 3.1.

3.2.3 *De novo* protein design

In contrast to mutation effect prediction, the approaches for *de novo* (literally "of new") protein design aim to generate novel proteins that satisfy specified constraints. For instance, ProT-VAE combines a Transformer pre-trained on protein sequences with a lightweight, adaptable autoencoder to generate novel protein sequences possessing desired properties (Sevgen et al., 2023). RFdiffusion is a recent approach that enables the generation of novel protein backbones (Watson et al., 2022). At its core, the model employs a deep learning architecture pre-trained on protein folding and learns to produce novel protein structures by diffusing existing ones from random noise. Similarly, FoldingDiff generates novel backbones by reconstructing native ones that have been corrupted by random twisting (Wu et al., 2022a). In our work, we do not further discuss *de novo* design methods, as our primary focus is on accurately improving the existing protein staphylokinase.

3.3 Analysis of big protein data

Proteins of higher importance have been studied more extensively than others. As a result, the datasets of proteins tend to cover the space of existing proteins in a highly non-uniform manner. This, along with other sources of biases in protein data, necessitates large-scale comparison and clustering of proteins to analyze

and prepare datasets for efficient machine learning and fair evaluation. While in traditional deep-learning domains such as computer vision or natural language processing, analysis and pre-processing can be achieved with the help of many non-expert annotators (Deng et al., 2009), the complexities of proteins are not easily understood by everyone. This strongly necessitates automated methods to filter, compare, and cluster large protein data. In this section, we discuss such approaches for proteins and their absence for protein–protein interactions.

3.3.1 Protein space

The most common way to compare two biological sequences is to align them and calculate the ratio of identical or similar characters at corresponding positions. However, this approach typically requires dynamic programming, which does not scale well for performing billions of comparisons. Steinegger and Söding (2017) have revolutionized protein chain analysis by developing the fast MMseqs2 algorithm for searching similar sequences. The core concept of the algorithm is constructing a database of k-mers from sequences and querying them in a highly optimized manner. The algorithm has enabled the clustering of all known protein sequences, resulting in the non-redundant database of protein chains UniRef (Steinegger and Söding, 2018).

Although fast search for similar sequences can be considered solved, it is not always sufficient for analyzing large-scale protein data. For instance, when searching for structural homologs of a protein, sequence similarity is typically not enough. As an example, proteins with highly different sequences can fold into almost identical shapes (van Kempen et al., 2022). Furthermore, multi-domain proteins have a highly modular structure, which poses challenges for methods based on whole sequence comparisons (Draizen et al., 2022). These facts strongly necessitate methods that directly compare protein structures.

Traditional approaches for comparing protein structures rely on alignment. For example, one can convert two protein structures into point clouds of residues and align them to minimize root mean squared deviation (RMSD). The RMSD value can then serve as a measure of similarity. TM-score improves upon RMSD by measuring global fold similarity rather than local structural deviations, and by making the score length-independent (Zhang and Skolnick, 2004). The TM-score for two protein structures represented by the alpha-carbons of their residues is

calculated using the following formula:

$$TM\text{–}score = \frac{1}{L_Q} \max \sum_{k=1}^{N_a} 1/(1 + d_k^2/d_0^2), \tag{3.1}$$

where the maximum iterates over possible alignments of the alpha-carbons of two proteins. The L_Q value represents the number of residues in one of the structures, considered as the query, and N_a denotes the number of residues being matched. Additionally, d_k is the distance between the alpha-carbons at the aligned position k, and d_0 is a constant that normalizes the distances to achieve length-independence of the score. The score varies between 0 and 1, with the value of 1 corresponding to the alignment of two identical structures.

Clearly, alignment-based structural similarity methods face the same challenges as sequence alignment methods: they are not scalable to large-scale data analysis. Recently, van Kempen et al. (2022) proposed the Foldseek algorithm for single-domain protein structures, which offers performance similar to structural alignment-based methods while being at least 20,000 times faster. The primary concept behind Foldseek is converting a protein structure into a sequence based on its backbone geometry and subsequently utilizing MMseqs2 for fast searching.

3.3.2 Protein–protein interaction space

While scalable comparison methods have been successfully established for protein sequences and single-domain structures, fast comparison of protein–protein interfaces remains unsolved. In this section, we briefly overview the most advanced available methods. First, Gao and Skolnick (2010a) proposed the iAlign algorithm, building upon TM-score. The IS-score of iAlign for a query interface with L_Q residues and any other interface calculates their similarity score by finding an optimal alignment of their alpha-carbons such that:

$$IS\text{–}score = (S + s_0)/(1 + s_0), \tag{3.2}$$

where

$$S = \frac{1}{L_Q} \max \sum_{k=1}^{N_a} f_k/(1 + d_k^2/d_0^2) \tag{3.3}$$

and the maximum iterates over possible alignments, akin to TM-score. The N_a value represents the number of residues being matched, and d_k denotes the dis-

tance between the corresponding alpha-carbons at the kth position of the alignment. Finally, f_k provides the ratio of matching distance-based contacts at the kth position, and d_0 and s_0 are fine-tuned constants to ensure length-independence of the score. IS-score has the same interpretation as TM-score but measures the similarity between interfaces.

Similarly, Mirabello and Wallner (2018) proposed the InterComp algorithm, which aims to find an optimal geometric alignment of interfacial residues while also considering the evolutionary similarity of matched residues. PCalign improves upon the aforementioned methods by additionally incorporating physiochemical features of residues (Cheng et al., 2015). Recently, Shin et al. (2023) proposed the PPI-Surfer algorithm, which performs alignment at the level of surface patches describing the physicochemical properties of the interfaces. All of these approaches rely on computationally-expensive alignment procedures and, therefore, are not scalable for big data analysis. To our knowledge, the only alignment-free method developed for comparing protein–protein interfaces is PatchBag (Budowski-Tal et al., 2018). The method utilizes the bag-of-words model applied to local patches of protein surfaces converted to a vector representation. This method is, however, not publicly available.

CHAPTER 4

Staphylokinase design with state-of-the-art machine learning methods

In this chapter, we discuss the selection of promising mutations of staphylokinase (SAK) with the goal of increasing its thrombolytic activity while preserving its necessary properties of high stability, low immunogenicity and limited dimerization. First, we apply several state-of-the-art ML methods to evaluate the effects of all possible single-point mutations on these properties. Next, we design an algorithm to robustly combine the obtained results and identify a limited number of the most promising substitutions. Finally, we discuss the construction of desired affinity-increasing multi-point mutations, which are currently undergoing experimental validation in the wet lab.

4.1 Datasets of labeled protein–protein interactions

4.1.1 Staphylokinase mutants

While rich data are essential for successful application of machine learning, it is often very difficult to gather comprehensive mutational scanning datasets specific to a given protein property. For instance, acquiring experimental data on biophysical properties such as $\Delta\Delta G$ of protein binding can be both time-consuming and expensive, with a single data point potentially costing thousands of dollars. On

the other hand, developing a statistical understanding of intricate combinatorial phenomena like epistasis necessitates a large number of samples (Dallago et al., 2021; Miton and Tokuriki, 2016).

Laroche et al. (2000) conducted experimental mutagenesis of staphylokinase, resulting in over 200 single- and multi-point mutations labeled with their impact on the activity (i.e. rate of plasminogen activation) and immunogenicity (i.e. rate of undesired antigen binding) of the protein. In our study, however, we aim to enhance the native affinity of staphylokinase towards plasmin. The Loschmidt Laboratories have compiled multiple studies into a small dataset of 13 staphylokinase variants with corresponding binary labels indicating the enhancement or disruption of binding towards plasmin. With such data scarcity, training a reliable staphylokinase-specific machine-learning model for affinity increase is unfeasible. Nonetheless, the available data can still be employed for evaluation purposes.

4.1.2 SKEMPI2 dataset

While rich PPI-specific data are typically unfeasible, aggregating experimental results from numerous studies offers a promising solution. The SKEMPI2 dataset is a manually-curated collection of annotated mutations in protein–protein interfaces, representing a significant stride towards addressing the general data scarcity issue in PPI design (Jankauskaitė et al., 2019). The dataset encompasses 295 published studies, covering a diverse range of interactions, including complexes of antibodies with antigens and proteases with inhibitors. Each SKEMPI2 entry is essentially represented by (i) a protein complex derived from the Protein Data Bank (PDB), (ii) a list of introduced mutations, and (iii) an experimentally measured value denoting the observed effect on binding affinity. The vast majority of the latter values can be converted to $\Delta\Delta G$ measurements, which represents the primary contribution of the dataset.

SKEMPI2 contains a total of 7085 annotated mutations, originating from 348 PPIs in 345 PDB files. These entries are grouped into "hold out types" according to the structural similarity of interfaces and the sequence identity of underlying partners, allowing for natural train-test splitting. It is important to note that the dataset has certain biases, including the dominance of naturally disruptive mutations ($\approx$ 80%) and the overrepresentation of single-point mutations ($\approx$ 75%), limiting the potential understanding of multi-point mutational effects. Addition-

ally, the substitutions are not uniform, reflecting common practices of mutagenesis experiments. In particular, almost half of the single-point mutations are substitutions to alanine, corresponding to conventional alanine scanning experiments.

4.2 Pre-selection of single-point staphylokinase mutations

4.2.1 Mutation evaluation

The SKEMPI2 dataset can be straightforwardly utilized to train a $\Delta\Delta G$ predictor, which can then be applied to score SAK mutations for improved affinity. However, the screening of the entire space of possible mutants is unfeasible. Even selecting 10 interface positions for mutagenesis results in 20^{10}, or approximately 10 trillion, multi-point variants. Consequently, the space of considered substitutions must be carefully reduced to the most promising ones. It is important to guarantee that any potential high-affinity mutant maintains crucial drug properties, ensuring its viability as a potential thrombolytic agent. The properties include high stability (i.e. the protein can survive) and low immunogenicity (i.e. the protein is accepted by the human body). Additionally, the potential for the dimerization of SAK must be minimized to ensure its therapeutic activity. Therefore, before approaching a SKEMPI2-based $\Delta\Delta G$ predictor to enhance the affinity of SAK to plasmin, we preselect a limited number of the most reliable substitutions to combine. For this purpose, we employ several well-established methods that address all aforementioned properties.

To assess protein stability, we utilize the sequence of staphylokinase and its crystal structure in a free, unbound form (PDB code `2SAK`). First, we construct a position-specific scoring matrix (PSSM) and obtain the calculations from the HotSpot Wizard software. Next, we estimate the probabilities of amino acid substitutions in the sequence by a forward pass of the ESM-1v transformer (Meier et al., 2021). To take advantage of the available tertiary structure, we also utilize ProteinMPNN to redesign the SAK sequence based on its native backbone, obtaining a 20-class probability distribution for each position in the sequence (Dauparas et al., 2022). Additionally, we obtain similar probability predictions using MutCompute (Shroff et al., 2020). Please see Section 2.1.3 and Chapter 3 for the description of the individual methods.

The affinity between partners in a protein–protein interaction is closely related to the stability of the resulting complex structure. This allows utilizing the structure-based methods ProteinMPNN and MutCompute to approximate affinity by predicting the stability of the complex structure (PDB code `1BUI`). Using these models, we obtain probability predictions for all twenty amino acids at each residue in the context of bound microplasmin. We get predictions of dimer instability using exactly the same approach for the dimer structure (PDB code `1C78`). Finally, we get immunogenicity predictions from a specialized SAK-specific model trained at Loschmidt Laboratories on the corresponding immunogenicity data. The methodology is summarized in Table 4.1.

Method	**Description**	**Addressed properties**	**Reference**
PSSM	Evolutionary-based statistics	Stability	Beckstette et al. (2006)
HotSpot Wizard	Biochemical software pipeline	Stability	Sumbalova et al. (2018)
ESM-1v	Sequence-based self-supervised deep learning	Stability	Meier et al. (2021)
MutCompute	Structure-based self-supervised deep learning	Stability, affinity, dimerization	Shroff et al. (2020)
ProteinMPNN	Structure-based self-supervised deep learning	Stability, affinity, dimerization	Dauparas et al. (2022)
PLS	SAK-specific machine learning	Immunogenicity	Loschmidt Laboratories

Table 4.1: Methods applied to score the single-point mutational space of staphylokinase.

4.2.2 Mutation selection

The obtained machine-learning predictions and software calculations lead to a vast amount of information which is not obvious how to systematically combine and reduce. Therefore, to achieve the pre-selection of a limited number of substitutions, we develop a consensus algorithm. We observe that the output of many protein-

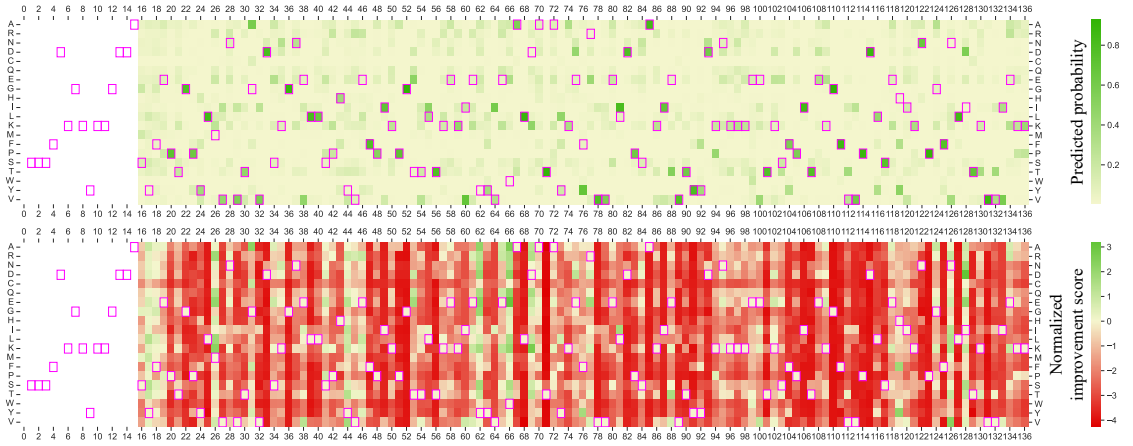

Figure 4.1: Staphylokinase stability matrix predicted by ProteinMPNN. The horizontal axis corresponds to the sequence of staphylokinase, while the vertical axis represents all the possible single-point mutations. The magenta boxes depict wild-type amino acids in a native SAK chain. Note that the residues 0-15 are missing in the crystal structure of `2SAK` and, therefore, do not have predictions. **Top)** The raw probabilities predicted by ProteinMPNN. **Bottom)** The corresponding normalized improvement scoring matrix obtained in step 9 of Algorithm 1.

design methods, including the ones in Table 4.1, can be represented as a matrix of 20 rows and L columns, where 20 is the number of natural amino acids and L is the length of a protein sequence (see Figure 4.1 Top for an example). We base the algorithm on two principles. First, we focus on whether substitutions improve upon native amino acids, and on the magnitude of the improvements, rather than how "good" the substitutions absolutely are. Second, we prioritize precision over recall. In other words, we do not aim to retrieve all favorable mutation candidates but the selected ones must be reliable.

The proposed method is detailed in Algorithm 1. The primary input for the algorithm is a wild-type protein sequence $\mathbf{w}$ and a collection of mutation matrices $\{\mathbf{M}_i\}_{i=1}^{n}$ such that their columns indicate any kind of scores of substitutions at each position. Additionally, each matrix must be associated with a property p_i that it evaluates (in our case the stability, affinity, dimerization, or immunogenicity), and each property must be assigned a sign g_i based on whether it is desired or not. Further, the algorithm requires assigning weights to all the matrices and properties, denoted as $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, respectively. The weights can all be set to one, manually adjusted according to expert understanding, or fitted to labeled data using simple linear or logistic regression, depending on the nature of the labels. The algorithm's output is a consensus set of the most promising single-point sub-

stitutions. The procedure can also be viewed as a robust filtering of the space of single-point mutations.

The algorithm consists of four primary steps. First, each matrix is transformed into a **normalized improvement scoring matrix**. This is accomplished by standardizing each matrix $\mathbf{S}$ using its standard deviation $\sigma^2_{\mathbf{S}}$, which is estimated from all its entries (i.e. a sample of all available scores). It is important to note that estimating the mean is not necessary, as standardization is directly followed by subtracting the values corresponding to wild-type amino acids in each column. This results in the conversion of normalized scores to normalized improvement scores. If the matrix is stochastic (i.e. the columns are probability vectors), it is first converted to logarithmic space to operate on log-likelihoods rather than raw probabilities. This implies that the obtained improvements are, in fact, normalized log-odds ratios well-established in mutation effect prediction (Riesselman et al., 2018). Additionally, if the matrix represents a property that must be minimized rather than maximized (e.g. dimerization in our study), the signs of its values are preliminarily flipped according to the provided sign g_i.

In the second step, the normalized improvement scoring matrices are averaged based on the properties p_i they represent and assigned weights α_i to generate **property scoring matrices**. For instance, if the algorithm is provided with five matrices representing protein stability and two matrices corresponding to immunogenicity, these two groups are averaged separately using the assigned matrix weights. Similarly, the property matrices are averaged in accordance with property weights β_i to produce a **final scoring matrix**. The final fourth step involves filtering the entries of the final matrix by selecting only non-disruptive (i.e. those with non-negative final normalized improvement scores) and non-identity (i.e. those that are not trivial substitutions to themselves) mutations.

4.3 Construction of multi-point staphylokinase mutations

After the space of all possible substitutions is reduced to a limited number of reliable ones with Algorithm 1, we can finally focus on the optimization of SAK's affinity to plasmin. Since a significant enhancement of protein function can be seldom achieved through a single-point mutation alone, we are interested in the

Algorithm 1: Consensus selection from single-point mutation matrices

Input: Wild-type protein sequence $\mathbf{w} \in \{1, \ldots, 20\}^L$, mutation matrices $\{\mathbf{M}_i\}_{i=1}^n$ ($\mathbf{M}_i \in \mathbb{R}^{20,L}$), property labels $\mathbf{p} \in \{1, \ldots, m\}^n$, property signs $\mathbf{g} \in \{-1, 1\}^m$, matrix weights $\boldsymbol{\alpha} \in \mathbb{R}^n$ and property weights $\boldsymbol{\beta} \in \mathbb{R}^m$.

Output: Non-disruptive non-identity single-point mutations $M \subset \{1, \ldots, 20\} \times \{1, \ldots, L\}$.

/* Get normalized improvement scoring matrices */

for $i \leftarrow 1$ to n **do**
  $\mathbf{S} \leftarrow \mathbf{M}_i$
  if $\mathbf{S}$ is stochastic **then**
    $\mathbf{S} \leftarrow \log \mathbf{S}$ // Convert probability to score
  $\mathbf{S} \leftarrow g_i \mathbf{S}$ // Invert scores if necessary
  $\sigma^2_{\mathbf{S}} \leftarrow stddev(flatten(\mathbf{S}))$
  for $r \leftarrow 1$ to 20, $c \leftarrow 1$ to L **do**
    $s_{r,c} \leftarrow (s_{r,c} - s_{w_c,c})/\sigma^2_{\mathbf{S}}$ // Standardize and center at wild types
  $\mathbf{S}_i \leftarrow \mathbf{S}$

/* Get property scoring matrices */

for $j \leftarrow 1$ to m **do**
  $I \leftarrow \{i \in \{1, \ldots, n\} \mid p_i = m\}$
  $\mathbf{P}_j \leftarrow \frac{1}{\sum_{i \in I} \alpha_i} \sum_{i \in I} \alpha_i \mathbf{S}_i$

/* Get final scoring matrix */

$\mathbf{F} \leftarrow \frac{1}{\sum_{j \in 1}^m \beta_j} \sum_{j=1}^m \beta_j \mathbf{P}_j$

/* Select non-disruptive non-identity mutations */

$M \leftarrow \{(r, c) \in \{1, \ldots, 20\} \times \{1, \ldots, L\} | f_{r,c} \geq 0 \wedge r \neq w_c\}$

return M

construction of affinity-increasing multi-point mutants. Referring to Figure 3.1, we identify GeoPPI as the best available multi-point $\Delta\Delta G$ predictor (Liu et al., 2021). However, the original source code only includes a model trained on a single-point part of the SKEMPI2 data. Therefore, we re-train the model on the whole SKEMPI2 dataset.

First, we preprocess the dataset to minimize the inherent biases and achieve indicative evaluation. In the first step, we balance the data by augmenting each mutation with a reversed one, flipping the $\Delta\Delta G$ values. Then, we leave out 6% of the data as a test set by selecting mutations from the `3BT1_A_U`, `2B11_A_B`, and `1KBH_A_B` complexes, which are likely to be most similar to SAK-microplasmin. While the first chosen interaction (`3BT1_A_U`) involves urokinase, i.e. another plas-

minogen activator, the other two (2B11_A_B, 1KBH_A_B) are selected as the best SAK-microplasmin matches by the iAlign and InterComp tools respectively (see Section 3.3.2). We partition the remaining data using a stratified group 5-fold split, ensuring a balanced distribution of $\Delta\Delta G$ labels across folds, with groups determined by the "hold-out types" in the SKEMPI2 dataset. After replacing the native random forest regressor with XGBoost (Chen and Guestrin, 2016), we select the best hyperparameters through cross-validation. This yields the best model with a mean RMSE of 2.21 on $\Delta\Delta G$ predictions. While this cannot be directly compared to the published GeoPPI performance because of the different data split, the obtained value is in the expected range according to the publication. Finally, the model demonstrates satisfactory performance on the test SAK-related data points, with an RMSE of 0.91 and a Pearson correlation coefficient of 0.46. After we re-fit the model to the whole dataset, we evaluate its performance on $\Delta\Delta G$ inference on 13 labeled mutations of staphylokinase. This independent test suggests the crucial effect of re-training, as visualized in Figure 4.2 A.

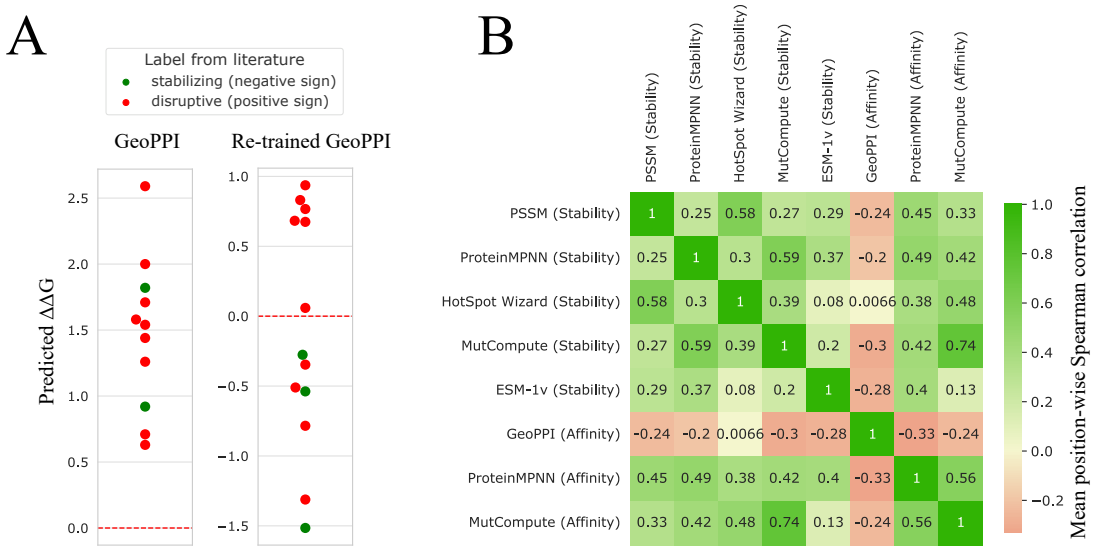

Figure 4.2: Staphylokinase-specific evaluation of GeoPPI. A) Evaluation of the GeoPPI models on an independent set of 11 single-point and 2 double-point staphylokinase mutations. The original GeoPPI model, trained on single-point data, predicts all mutations as disruptive (positive $\Delta\Delta G$), while the re-trained model identifies favorable mutations (by correctly predicting negative $\Delta\Delta G$). **B)** Pairwise mean column correlations of normalized improvement scoring matrices for 20 SAK interface positions reveal distinct scoring patterns of GeoPPI compared to the methods used for mutation pre-selection. Note also how several dissimilar methods, such as PSSM (sequence-based evolutionary statistics) and ProteinMPNN (structure-based evolutionary-agnostic deep learning), remarkably exhibit a high positive mutual correlation.

Whilst the validation of re-trained GeoPPI demonstrates promising scoring capabilities, its further application reveals some concerning behavior. First, we observe that the model is not robust to minor, seemingly negligible changes in the SAK structure. For example, subtle side-chain optimization with the force-field software FoldX (Delgado et al., 2019) often leads to opposite-sign $\Delta\Delta G$ predictions by GeoPPI. Additionally, to directly compare GeoPPI to the previously selected substitution pre-selection methods, we apply GeoPPI to score all single-point mutations and obtain a corresponding mutation matrix. Figure 4.2 B shows that the GeoPPI matrix does not positively correlate with the other ones. This discrepancy could either indicate that its predictions are orthogonal, introducing new information, or that they are unreliable. Biochemistry experts from Loschmidt Laboratories notice that the top-ranked amino acids at 18 out of 20 positions are aromatic residues (W, Y, F, or H), which are bulky and often risky to mutate to. This observation significantly undermines the reliability of GeoPPI, even for single-point substitutions. As a result, for the evaluation of multi-point mutants, we use a traditional non-ML method Affilib (see Section 2.1.3).

4.4 Results

To select the final mutation candidates for wet-lab experiments, we begin by choosing the most promising single-point substitutions, as discussed in Section 4.2. We combine the methods from Table 4.1 using Algorithm 1. We set the weights according to the contemporary understanding of SAK properties by experts at Loschmidt Laboratories. The procedure results in the selection of 39 favorable single-point mutations out of all $20 \times 136 = 2720$ possible ones. 17 of them correspond to different substitutions at two interface positions, indicating their high potential for affinity increase. Additionally, several from the other 22 mutations exhibit very high final normalized improvement scores, suggesting themselves as promising candidates. The simplicity of Algorithm 1 allows explaining the chosen mutations by backtracking through the corresponding intermediate matrices. For instance, although the mutation of glutamic acid at position 46 to serine is suggested as an affinity-improver by both ProteinMPNN and MutCompute, it has very low predictions for free SAK stability and is, therefore, not included in the final pool. We create a simple interactive website that visualizes the selection,

including intermediate steps for analysis[3]. As discussed in Section 4.3, the pre-selected single-point mutations are further combined with Affilib to construct several dozens of multi-point mutations.

The final obtained variants represent an effort to increase the thrombolytic activity of staphylokinase by the application of the best currently-available machine-learning methods. At the moment of writing, the selected candidates are undergoing the stage of experimental wet-lab validation. From the broader perspective, our case study illustrates that while the state-of-the-art machine learning tools for protein design enable robust pre-selection of single-point protein substitutions, a reliable method for mutational PPI design is still critically lacking. This fact motivates the remainder of the thesis. We aim to establish a machine-learning model capable of reliably scoring single- and multi-point mutations for future rounds of SAK design and other studies on protein–protein interactions.

[3]Protected link: https://anton-bushuiev.notion.site/Combination-of-mutation-matrices-for-SAK-design-8d24de0292634db9bf08909be0030277

CHAPTER 5

Preparing the datasets of protein–protein interactions

In the previous chapter, we have identified the unreliability of the state-of-the-art machine-learning models for the design of protein–protein interactions. We argue that the primary cause of this unreliability lies in their dependency on the small annotated data represented by SKEMPI2. While we expand on this argument in the following Chapter 6, here we discuss the analysis and preparation of a large unannotated dataset of protein–protein interactions extracted from the whole Protein Data Bank. We find that there is a weak contemporary understanding of the composition and biases of the existing data of this kind. Consequently, we conduct an analysis that reveals significant limitations of their current utilization, as exemplified by the typical validation of models on test data that is highly similar to the training data.

5.1 Datasets of unlabeled protein–protein interactions

5.1.1 Protein Data Bank

The Protein Data Bank (PDB) is a central resource for machine learning from protein structures, consisting of entries obtained via X-ray crystallography, NMR spectroscopy, or cryo-electron microscopy, submitted by biologists and biochemists worldwide (Berman et al., 2000). With over 200,000 structures, the large subsets

of the database have served to train the models discussed in Chapter 3. Each PDB entry contains the 3D coordinates of one or more arranged molecules (primarily proteins) and is associated with a four-letter code. For example, `2SAK` corresponds to the file which contains a crystal structure of staphylokinase.

It's important to recognize that PDB contains non-trivial biases. Proteins of significant practical interest have been studied more extensively, resulting in their over-representation (Draizen et al., 2022). For instance, querying Protein Data Bank for "Hemoglobin" returns nearly 600 structures that are indistinguishable for typical machine learning purposes but, for example, arise from different species. Not only is PDB non-uniform, but it's also highly redundant. On average, a protein is represented over four times in the database (Burra et al., 2009). On the other hand, Skolnick et al. (2012) showed that PDB is complete for single-domain structures.

Similar conclusions have been drawn regarding the protein–protein interactions represented in PDB. Through the application of iAlign to a representative sample of 1,519 PPIs, Gao and Skolnick (2010b) demonstrated that PDB may be complete for protein–protein interactions by containing 90% of the possible native interfaces. Furthermore, the study revealed that there may be approximately 1,000 distinct interaction types, emphasizing the extreme redundancy of the space. Besides that, about 80% of protein–protein interactions were shown to form a dense network with a diameter of seven, meaning that the majority of interfaces have highly homologous structures. These results offer promise in the comprehensive statistical understanding of the vast human interactome, which is represented by over 100,000 PPIs (Geng et al., 2019b).

5.1.2 Database of Interacting Protein Structures

The Database of Interacting Protein Structures (DIPS) was created with the goal of extracting all protein–protein interactions from the Protein Data Bank (Townshend et al., 2019). As far as we know, this is currently the only dataset of its kind and we review it below. DIPS was initially constructed for the task of rigid-body protein–protein docking by identifying all dimeric structures in PDB and extracting their interfaces. In the case of DIPS, an interface was defined as a set of amino acids that have at least one non-hydrogen atom within 6Å of a non-hydrogen atom belonging to the other partner (hydrogen atoms are typically not observed in experimental

structures). The extracted interfaces were then filtered to meet four criteria:

1. The underlying structure is the first model in a file and solved using X-ray crystallography or cryo-electron microscopy at better than 3.5Å resolution.
2. The underlying chains are longer than 50 amino acids.
3. The buried surface area (BSA) of the interface is $\geq 500\text{Å}^2$.
4. None of the partnering proteins has over 30% sequence identity when aligned to any protein in the DB5 docking dataset.

While (1) removes low-quality or trivially repeated entries and (2) ensures the partners are proper proteins rather than peptides, (3) guarantees the selected interfaces are large "enough" to form interactions. Condition (4) is task-specific and designed to prevent data leakage with respect to the test set used in the study. The described pre-processing resulted in the collection of 42,826 PPI structures. Morehead et al. (2021) have further enriched the data with multiple kinds of per-residue features resulting in DIPS-Plus.

5.2 Fast algorithm to compare protein–protein interactions

Despite the active utilization of DIPS for machine learning, its composition and biases have never been analyzed due to the absence of scalable methods to compare protein–protein interactions. In this section, we present the iDist algorithm which enables large-scale analysis of protein–protein interface data.

5.2.1 Motivation

Despite the widespread use of DIPS in machine learning applications, its composition and inherent biases have never been addressed (Ketata et al., 2023; Wang et al., 2023; Ganea et al., 2021; Morehead et al., 2021; Townshend et al., 2019). Likewise, the quality of the train-test splits employed has never been examined. By detecting strong biases in DIPS (see Section 6.2) and taking into account important facts about the composition of the underlying Protein Data Bank, we pose three fundamental questions:

Q1) What is the exact composition (i.e. redundancy, connectivity, and completeness) of DIPS?

Q2) What is the quality of existing data splits of DIPS?

Q3) What is the relationship between PPIs in DIPS and SKEMPI2?

Answering the first question (Q1) is essential to gaining a comprehensive understanding of the applicability of the dataset, and to interpret the results derived from training. Firstly, learning from redundant subsets of PDB has been demonstrated to hinder machine learning performance by introducing biases towards overrepresented proteins (Shroff et al., 2020). Consequently, the redundancy of DIPS may cause machine learning to rely on biases rather than learning biochemical features. Secondly, discovering that DIPS is highly connected may suggest that creating leakage-free splits of DIPS is extremely challenging or impossible. Likewise, a high incompleteness of DIPS may imply that training on the dataset may not generalize to many other protein–protein interactions of practical interest. Since train-test splits of DIPS have never been analyzed, answering the second question (Q2) may provide a better understanding of the practical applicability of the models trained on them. Additionally, the answer may indicate which data split, if any, should be considered a standard benchmark. In general, it is crucial to avoid having identical or highly similar entries in the training and test parts of the data. Finally, answering the third question (Q3) is crucial for understanding the potential of transfer learning from the large-scale, unannotated DIPS to the small $\Delta\Delta G$-annotated SKEMPI2 dataset for protein design.

Answering all three questions can be reduced to the comparison of protein–protein interfaces. Indeed, (Q1) can be majorly answered by clustering the dataset, and (Q2, Q3) only require measuring distances between the sets of interfaces. Additionally, the primary interest of comparison lies in the detection of highly related protein–protein interfaces rather than differentiation between various levels of similarity. This is analogous to the problem of near-duplicate detection in computer vision (Thyagharajan and Kalaiarasi, 2021). While image data can be affected by issues such as copied or repeated images, minor variations in conditions or rotations, the protein data can be similarly affected by small mutations, conformational changes, or arbitrariness in the coordinate systems used in PDB files. The detection of near-duplicate protein–protein interfaces can further serve

to remove inherent biases from PPI data, reduce computational time by creating a representative subset of cleaned data, and facilitate fair evaluations by constructing appropriate, non-leaked data splits.

Furthermore, to be efficient, the desired method to compare protein–protein interactions should be scalable to analyze the described large-scale DIPS. However, the available methods discussed in Section 3.3 are not scalable because they require pairwise alignment. For example, iAlign is relatively fast, enabling approximately three comparisons per second on a single CPU. Nonetheless, analyzing the entire DIPS dataset would require $42,826^2 \approx 1,8$ billion comparisons, resulting in nearly two months of computational time when utilizing 128 CPUs in parallel. Note that the time complexity of pairwise comparison of N interfaces may be decomposed as $O(r \cdot N) + O(c \cdot N^2)$, where r and c are constants representing the time needed to suitably represent a PPI and the time to compare two representations, respectively. Available alignment-based algorithms, therefore, have high c and relatively low r. However, to enable large-scale comparison, one needs to, conversely, significantly reduce c, which can be achieved by increasing the representation time r.

5.2.2 iDist algorithm

In this work, we develop a simple scalable algorithm to measure the distance between protein–protein interfaces, which we further refer to as iDist. We reason that by embedding PPIs in low-dimensional vector space we can minimize the pair-wise complexity c by reducing comparison to a simple well-optimized distance measure on vectors d. Nevertheless, to be effective, representations should satisfy certain properties. First, the algorithm should be invariant to the ordering of partnering chains in PPIs. Second, it should be invariant to the ordering of residues in chains, also known as topology independence. Importantly, the representations should be roto-translationally invariant, guaranteeing that for any interface $\mathcal{I}$ and any rigid transformation $f \in SE(3)$, the given distance measure d satisfies $d(\mathcal{I}, f(\mathcal{I})) = 0$. This property can be seen as implicit alignment. Lastly, the representations should be as rich as possible to effectively capture differences between interfaces.

Algorithms 2 and 3 outline iDist, a fast method for comparing protein-protein interfaces while satisfying the described properties. Algorithm 2 details the con-

Algorithm 2: iDistEMBED

Input: Protein–protein interface $\mathcal{I}$ of n residues.
Output: vector representation of the interface $\mathbf{z}_{\mathcal{I}}$.

/* Get coordinates, features, and binary partner information of residues */
1 $\mathbf{X} \in \mathbb{R}^{n,3}, \mathbf{F} \in \mathbb{R}^{n,d}, \mathbf{p} \in \{0,1\}^n \leftarrow get\_residues(\mathcal{I})$
/* Embed residues */
2 **for** $i \leftarrow 1$ to n **do**
3 $\quad J_{intra} \leftarrow \{j \in \{1,\ldots,n\} \mid p_j = p_i \wedge j \neq i\}$
4 $\quad J_{inter} \leftarrow \{j \in \{1,\ldots,n\} \mid p_j \neq p_i\}$
5 $\quad \mathbf{m}_{intra} \leftarrow \frac{1}{|J_{intra}|} \sum_{j \in J_{intra}} \mathbf{f}_j \cdot e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\alpha}}$
6 $\quad \mathbf{m}_{inter} \leftarrow \frac{1}{|J_{inter}|} \sum_{j \in J_{inter}} \mathbf{f}_j \cdot e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\alpha}}$
7 $\quad \mathbf{h}_i \leftarrow \frac{1}{2}\mathbf{f}_i + \frac{1}{4}\mathbf{m}_{intra} - \frac{1}{4}\mathbf{m}_{inter}$
/* Embed interface */
8 $J_0 \leftarrow \{j \in \{1,\ldots,n\} \mid p_j = 0\}$
9 $J_1 \leftarrow \{j \in \{1,\ldots,n\} \mid p_j = 1\}$
10 $\mathbf{z}_{\mathcal{I}} \leftarrow \frac{1}{2}(\frac{1}{|J_0|} \sum_{j \in J_0} \mathbf{h}_j + \frac{1}{|J_1|} \sum_{j \in J_1} \mathbf{h}_j)$
11 **return** $\mathbf{z}_{\mathcal{I}}$

Algorithm 3: iDist

Input: Two protein–protein interfaces $\mathcal{I}$ and $\mathcal{J}$.
Output: Distance ≥ 0.
1 $\mathbf{z}_{\mathcal{I}} \leftarrow$ iDistEMBED($\mathcal{I}$)
2 $\mathbf{z}_{\mathcal{J}} \leftarrow$ iDistEMBED($\mathcal{J}$)
3 **return** $\|\mathbf{z}_{\mathcal{I}} - \mathbf{z}_{\mathcal{J}}\|_2$

version of a protein-protein interface $\mathcal{I}$ into a vector representation $\mathbf{z}_{\mathcal{I}}$. Initially, the features of the interface $\mathbf{X}, \mathbf{F}$ and $\mathbf{p}$ are extracted. The residue coordinates $\mathbf{X}$ are determined by the positions of the C_α atoms. Next, the residue vector features $\mathbf{F}$ are initialized with simple 20-dimensional one-hot encodings of amino acids. We also experiment with ESM-1 features (see Chapter 3), but get slightly lower performance. We observe that the reduced performance is attributed to ESM-1 biasing the comparison towards entire chains rather than interfaces. Subsequently, each residue is associated with a binary label based on the arbitrarily chosen order of interacting chains, forming the vector $\mathbf{p}$.

In the following step, detailed in lines 2-7, the hidden representations $\mathbf{h}_i$ for each residue i are constructed. The key idea behind the approach is that each

residue receives messages from other residues within the same chain J_{intra} as well as from the other chain J_{inter}. Inspired by (Dauparas et al., 2022), the messages are represented by exponential radial basis functions (with α set to 16). Each node averages intra- and inter-messages into contact patterns $\mathbf{m}_{intra}$ and $\mathbf{m}_{inter}$. The representation $\mathbf{h}_i$ is then obtained by averaging the difference between $\mathbf{m}_{intra}$ and $\mathbf{m}_{inter}$ (we reason that the difference may capture the nature of complementarity of biochemical interactions), followed by averaging with the initial features.

Lastly, in steps 8-11, the interface representation $\mathbf{z}_{\mathcal{I}}$ is derived by averaging the hidden features across chains and then across the interaction. As described in Algorithm 3, iDist then simply computes the Euclidean distance between two representations $\mathbf{z}_{\mathcal{I}}$ and $\mathbf{z}_{\mathcal{J}}$ to compare two interfaces $\mathcal{I}$ and $\mathcal{J}$.

5.2.3 Validation of the proposed iDist algorithm

To evaluate the performance of the proposed iDist algorithm, we benchmark it against the alignment-based IS-score of iAlign (Gao and Skolnick, 2010a) described in Section 3.3. This approach is well-justified by common practices. For example, Foldseek was evaluated by the comparison with TM-score (van Kempen et al., 2022). As IS-score is the adaptation of the latter to protein–protein interfaces, the benchmarking of iDist against the IS-score is a natural choice. For the evaluation, we randomly sample 100 PDB codes from DIPS and select all PPIs from the corresponding files, resulting in 1646 interfaces.

We compute all $1646 \times 1646 = 2,709,316$ pairwise distances between sampled interfaces with both iDist and iAlign. The pairwise computation of IS-score on 128 CPUs in parallel took 2 hours, consistent with the estimate mentioned above, while the same calculation with iDist took 15 seconds. Figure 5.1 A displays the joint histogram, indicating a significant correlation between the pairwise comparisons with iDist and iAlign ($\rho_{Pearson} = -0.38$, $\rho_{Spearman} = -0.36$). The discrepancy between the methods increases as the IS-scores decrease, implying that the variance of iDist increases when the interfaces do not align well. This observation is in line with the general observation by LeCun and Misra (2021): “To paraphrase Leo Tolstoy’s Anna Karenina: “Happy families are all alike; every unhappy family is unhappy in its own way.” This applies to any family of high-dimensional objects, it seems.” Indeed, unalignable interfaces correspond to a wide range of distances, whereas the distance spectrum narrows as the alignment improves. Figures 5.1

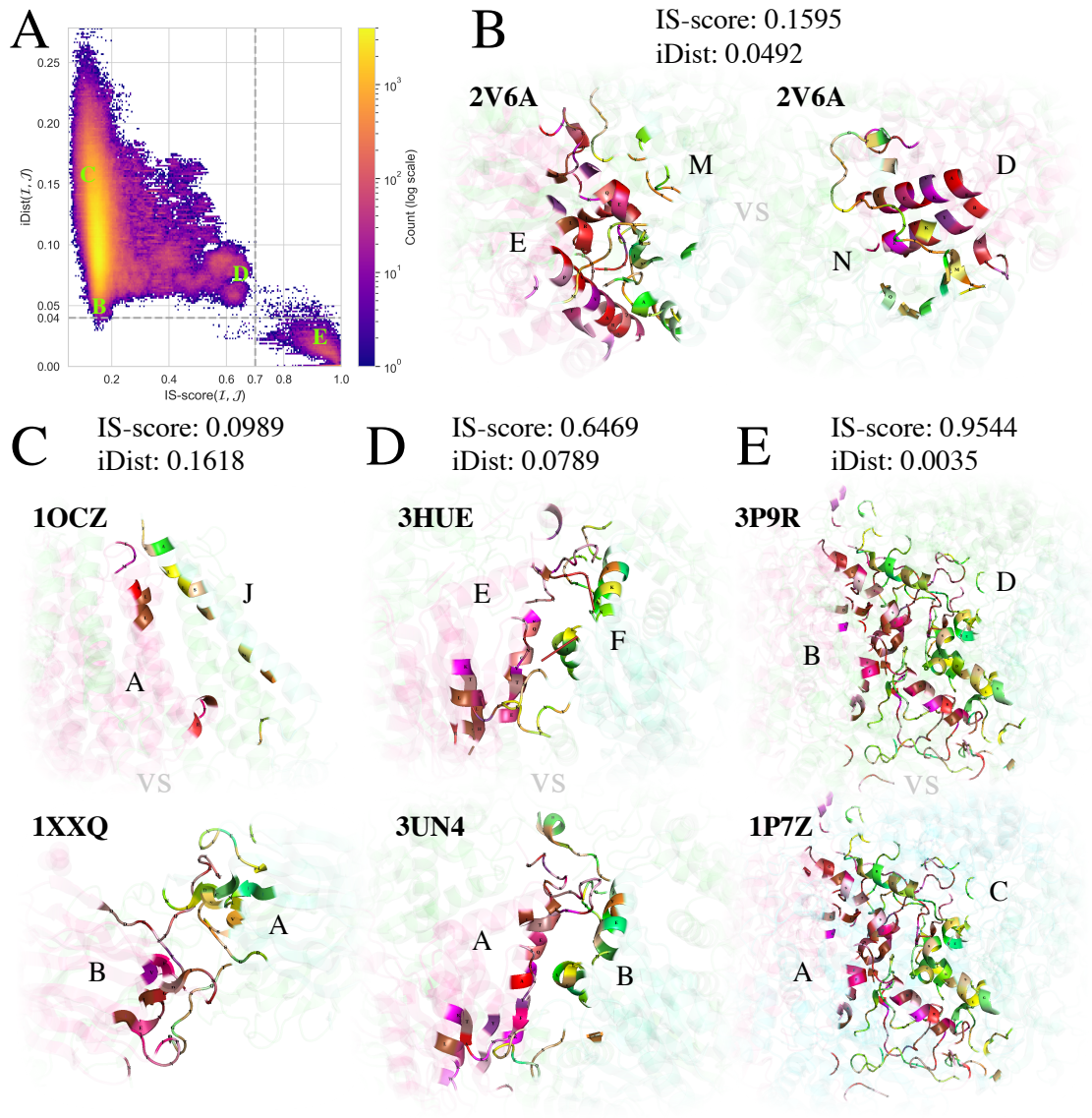

Figure 5.1: Performance of iDist. **A)** Joint log-scale histogram displaying pair-wise IS-scores and iDist values of 1646 sampled PPI interfaces. The IS-score varies between 0 and 1, with high values corresponding to well-alignable interfaces (1 for identical interfaces) and low values corresponding to poorly-alignable ones. The iDist varies between 0 and 0.3 with high values corresponding to structurally-distant interfaces and low values corresponding to similar ones (0 for identical interfaces). Figures **(C, D, E)** depict samples from regions where the methods correlate, while **(B)** shows an example of disagreement. Each figure displays two interfaces colored by amino acid types, with one protein's palette in reddish hues and the other one in greenish. **B)** Ambiguous comparison. The IS-score corresponds to the expected value of the alignment of two random PPIs, while iDist suggests high similarity due to the identity of several fragments of chains M and N (note the ε-like green shape and its further continuation) and similar composition of helices belonging to E and D (similar combination of reddish colors). In fact, the two interfaces represent different interaction modes of the same two chains in a big symmetric complex. **C)** Unrelated interfaces. **D)** Interfaces on the edge of being considered near-duplicates. The interactions are obviously related, but the geometry and primary structure differ at every local fragment. **E)** Near-duplicates.

B-E illustrate the examples of comparison.

Next, we evaluate the retrieval capabilities of iDist by taking IS-score values as the ground-true labels of relevance. As such, we select a threshold of IS-score $\tau_{IS\text{–}score}$ and define comparisons with IS-score greater than $\tau_{IS\text{–}score}$ as relevant items, or hits. Then, we select a similar threshold for iDist τ_{iDist} and use standard information retrieval metrics to evaluate the performance – precision and recall. As, in this work, we are primarily interested in near-duplicate detection, we set $\tau_{IS\text{–}score} = 0.7$ which corresponds to a separate mode in a marginal distribution of IS-score in Figure 5.1 A (see also the difference between Figures 5.1 B, D and Figure 5.1 E). By fixing $\tau_{iDist} = 0.04$ and considering the $(\mathcal{I}, \mathcal{J})$ interface pair a hit if $iDist(\mathcal{I}, \mathcal{J}) < \tau_{iDist}$, we obtain a fast near-duplicate detector with 0.99% precision and 0.97% recall with respect to IS-score (see the dashed lines in Figure 5.1 A).

5.3 Data analysis and preparation

In this section, we apply the proposed iDist algorithm to answer the questions raised in Section 5.2.1. We begin by answering (Q1) with the analysis of the composition of DIPS. Then, we provide an answer for (Q2) demonstrating that the existing splits suffer from data leakage. Finally, we show that DIPS and SKEMPI2 are almost disjoint, answering (Q3). We conclude the section by constructing our own data splits with the aid of iDist.

5.3.1 DIPS is highly-connected, redundant and not complete

To analyze the composition of DIPS, we first calculate the pairwise distances between all PPIs using iDist. We construct a near-duplicate network of DIPS by connecting two interfaces if their distance is lower than $\tau_{iDist} = 0.04$, as discussed in the previous section. This results in a graph with 8.5K components, while the largest component comprises 36% of the interfaces. Increasing τ_{iDist} to 0.06 results in 84% of the interfaces forming a single component, indicating the high connectivity of the PPI space in DIPS. We then iteratively remove entries with near-duplicates (measured by the same $\tau_{iDist} = 0.04$). This reduces the size of DIPS to 22% of its initial size. For instance, the B-D interaction from `3P9R` shown in Figure 5.1 E has 79 near-duplicates from 33 PDB files, while the most abundant interface is instantiated in the interaction between chains C and D in `1YOV`, having

592 near-duplicates. These observations additionally suggest that DIPS is highly redundant.

We further analyze the relationship between PPIs in DIPS and those in SKEMPI2. We find that DIPS contains only 7 out of the 348 interfaces present in SKEMPI2, meaning that 98% of the interfaces did not pass one of the four filtering criteria described in Section 5.1.2. While the filtering source code of DIPS is not available, we hypothesize that condition 3 (BSA $\geq$ 500Å^2) is either erroneous or too strict for practical purposes beyond protein–protein docking. For example, calculating the buried surface area of the interfaces formed by chains B and C from 3SE3 and A, B from 4GON with the independent software *dr_sasa*, we obtain values higher than 500Å^2, while the interactions are not present in the DIPS file comprising entries that passed the filter (Ribeiro et al., 2019). Overall, the fact that DIPS and SKEMPI2 are almost disjoint indicates the incompleteness of the former.

The obtained results are in agreement with those by Gao and Skolnick (2010b). While searching for structurally-related interfaces in a smaller sample of PDB interfaces with iAlign led to the hypothesis that the space of PPIs in PDB is highly-connected, redundant, and close to complete, applying iDist to a contemporary large-scale DIPS similarly suggests that it is connected and redundant, however not complete due to its seemingly too strict construction. Additionally, the obtained order of interface redundancy (78%) interestingly agrees with the above-mentioned fact that an average protein is represented in PDB over four times.

5.3.2 Existent data splits do not measure generalization

Previous research conducted on machine learning from DIPS has not given much importance to data splitting and its analysis. In this study, we aim to address this gap by utilizing iDist to evaluate the quality of available splits. For the analysis, we first find the nearest neighbor of each test PPI in the training fold and consider it to be a leak if the distance falls below the near-duplicate threshold $\tau_{iDist} = 0.04$. Our findings indicate that the random split used in DIPS-Plus results in the test set with 88% of leaks (Morehead et al., 2021). Meanwhile, the EquiDock split, based on protein families of interacting partners, has 53% of leaks (Ganea et al., 2021; Ketata et al., 2023). This may pose a significant limitation on the validation and benchmarking of well-designed models. Figure 5.1 E illustrates a leakage example, with the top interaction being used for training and the bottom one for the testing

of EquiDock and DiffDock-PP. While the protein classification taxonomy used in the study is not specified, we speculate that the high leakage ratio may be caused by sequence-based splitting pitfalls discussed in Section 3.3, as family definitions are typically based on sequence similarity (Andreeva et al., 2014). Moreover, previous studies have demonstrated that even different proteins can form almost identical interactions, indicating that partner-level splitting may be insufficient (Gao et al., 2023).

Although we do not employ iDist to analyze the small space of protein–protein interactions in SKEMPI2, it is worth noting that the current methods of splitting mutations on these interfaces are often naïve, potentially leading to data leakage as well. For practical purposes, one is interested in estimating the performance of a $\Delta\Delta G$ predictor on unseen interfaces. This corresponds to the independent downstream application of the method to the screening of a particular protein–protein interaction such as staphylokinase–microplasmin. SKEMPI2 offers a clustering of interfaces specifically designed for this purpose, as discussed in Section 4.1.2. However, these annotations are often disregarded in favor of standard k-fold cross-validation on the level of mutations (Wang et al., 2020; Zhou et al., 2020a). Inspired by the fact that some complexes in SKEMPI2 are highly related, Liu et al. (2021) constructed their own alternative to the clustering of complexes provided in SKEMPI2, which is, however, not publicly available.

5.3.3 Constructed datasets

To achieve efficient deep learning from protein–protein interactions, we preprocess the DIPS and SKEMPI2 datasets. We clean the data using our near-duplicate detector and divide each dataset into two parts for training and validation. In this study, we do not require a conventional three-part split (training, validation, and test) as we explore the generalization between DIPS and SKEMPI2.

First, we clean the DIPS dataset by applying iDist with the standard threshold $\tau_{iDist} = 0.04$ to iteratively detect and remove near-duplicate entries, as described previously. We then divide the remaining data according to (Ganea et al., 2021). Note that after the cleaning, only the representative interfaces remain, and therefore, the split does not introduce data leakage as identified previously considering the raw data. In total, the training and test portions of DIPS contain 8675 and 497 protein–protein interfaces, respectively.

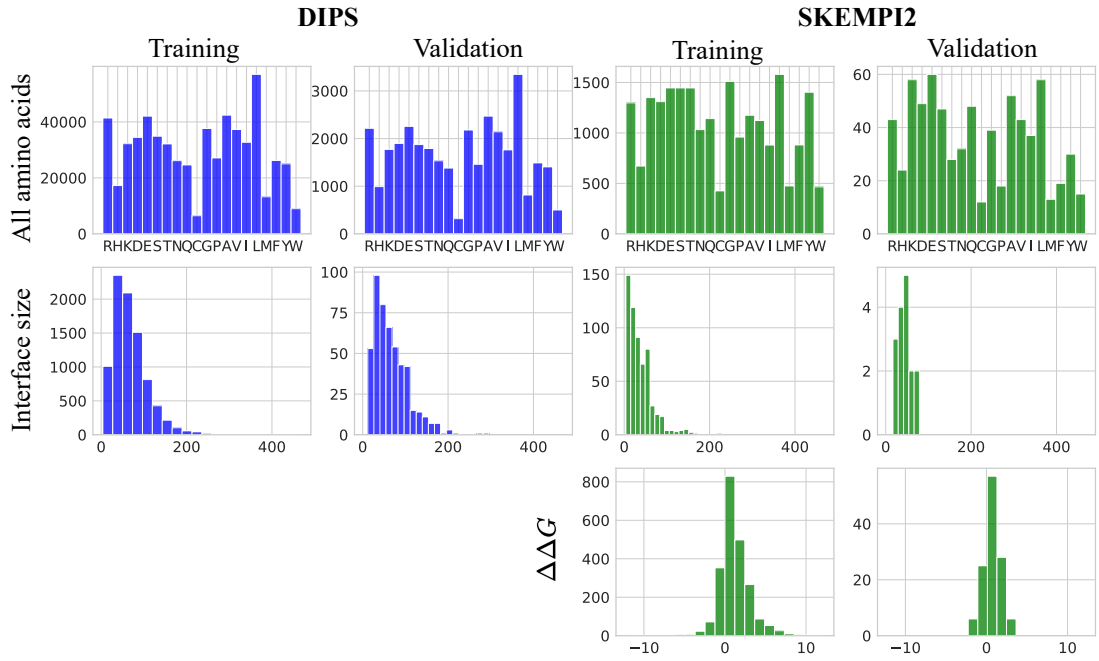

Figure 5.2: Statistics of constructed datasets. The first row of histograms displays the overall amino acid composition of the interfaces, while the second row shows the distribution of interface sizes, i.e. total numbers of residues. The third row illustrates the distributions of $\Delta\Delta G$ labels associated with all annotated interfacial mutations.

Next, we extract all protein–protein interfaces from SKEMPI2 that contain at least one annotated mutation, i.e. have at least one associated $\Delta\Delta G$ label. To ensure consistent data representation, we extract interfaces from SKEMPI2 according to the DIPS interface definition. Consequently, not all available annotated mutations are included in the obtained interfaces. Furthermore, in this study, we only consider single-point mutations, which further reduces the number of utilized labels. We divide the data based on the distribution of $\Delta\Delta G$ labels and the SKEMPI2 hold-out types that cluster interfaces into biologically and structurally related groups. Specifically, we apply a stratified group 5-fold split to the whole SKEMPI2 dataset, dividing all mutations into five parts with similar distributions of $\Delta\Delta G$ labels and complexes with the same hold-out types in separate groups. We then select four of the mutation groups to define the interfaces for the training part and use the remaining one for the validation part. In total, the training part of SKEMPI2 contains 592 interfaces and 2253 annotated single-point mutations, while the test part comprises 23 interfaces with 122 mutations.

Figure 5.2 displays the statistics of the resulting datasets. Interestingly, the amino acid composition of interfaces in the training and validation portions of

DIPS is nearly identical, despite the removal of data leakage. We attribute this phenomenon to the identified extreme connectivity of DIPS. In contrast to the high similarity between the training and validation portions of DIPS, SKEMPI2 exhibits a distinct distribution. Moreover, in line with our hypothesis regarding overly strict DIPS filtering for interfaces with large buried surface areas, SKEMPI2 contains interfaces consisting of a smaller number of residues on average.

CHAPTER 6

Self-supervised learning from protein–protein interactions

In recent years, deep learning has significantly transformed the field of protein science, offering solutions to several fundamental biochemical problems, such as protein folding and inverse protein folding (Jumper et al., 2021; Dauparas et al., 2022). However, despite active research in these areas, as well as, for example, *de novo* protein design or rigid-body docking, there has been limited focus on understanding the statistical principles governing protein–protein interfaces. In contrast, many problems related to protein–protein interactions, such as $\Delta\Delta G$ prediction or docking pose scoring, have been studied separately, undergoing a similar gradual development over more than a decade Geng et al. (2019b). The resulting task-specific models often suffer from instability or poor generalization due to the small sizes of their underlying datasets (Jin et al., 2023; Geng et al., 2019b), an issue encountered in our staphylokinase design case study described in Chapter 4.

Inspired by the successes of deep learning discussed in Chapter 3 and the availability of large-scale protein–protein interaction data analyzed in Chapter 5, we aim for the establishment of a foundational model for protein–protein interfaces. Specifically, we hypothesize that self-supervised training using a vast amount of unlabeled protein–protein interactions can lead to a general understanding of the statistical principles governing the interfaces. Consequently, a model trained in this manner could potentially generalize across various downstream problems, unifying task-specific approaches that tend to suffer from the constraints of lim-

ited annotated data.

In the subsequent sections, we present our approach, a self-supervised geometric deep learning model, which we refer to as PPIFORMER. We first discuss the data representation, model architecture, and masking-based training procedure. Given that the primary focus of our work is protein design, we also explain how the model can be adapted for the task of $\Delta\Delta G$ prediction. We conclude this chapter by demonstrating the proof of concept for the proposed approach. Specifically, we show that (i) PPIFORMER , by learning the biochemical principles governing protein–protein interactions, can generalize to statistically distinct interactions drawn from an independently collected dataset, and (ii) the representations emerging in PPIFORMER capture protein-design principles, enabling unsupervised $\Delta\Delta G$ scoring.

6.1 PPIFORMER

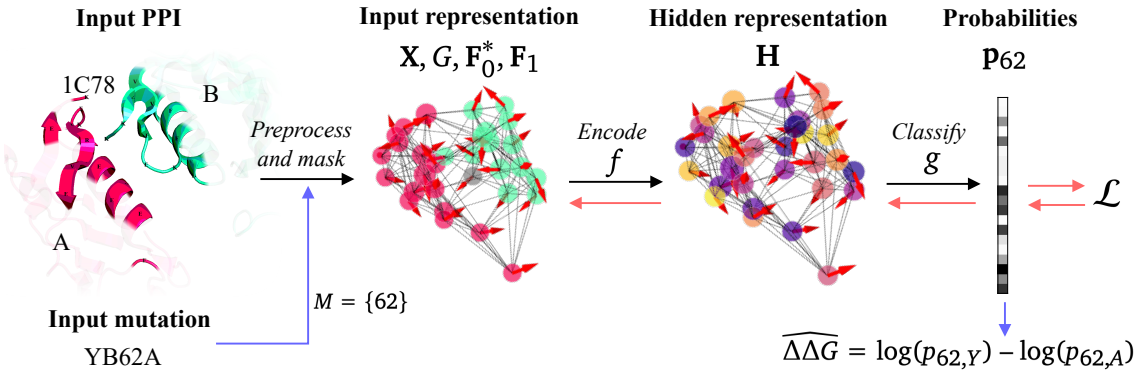

Figure 6.1: Training and inference of PPIFORMER. The black arrows in the pipeline depict the model's architecture, while the red arrows demonstrate the self-supervised training process for classifying missing amino acid side chains. A single training step begins with randomly sampling a protein–protein interaction (e.g. A-B from 1C78). After converting the interface into a point-cloud representation, the features defining the side chain of a randomly chosen node (e.g. 62 from chain B) are masked (shown by the grey point). The model subsequently learns to classify the masked type of the amino acid by acquiring an appropriate hidden representation of the whole interface. The blue arrows illustrate the masked-marginals regime of $\Delta\Delta G$ inference. To predict the mutational effect of substituting tyrosine (Y) with alanine (A) at protein position 62, the corresponding position is masked, and the probabilities are predicted using the trained model. Finally, the $\Delta\Delta G$ is estimated according to the derived probabilities.

6.1.1 Data representation

In order to apply deep learning to protein–protein interfaces it is crucial to select an appropriate representation. In this study, we represent an interface as an oriented point cloud of residues, meaning that the smallest unit considered is an entire amino acid. We believe that abstracting residues from their internal atomic structure allows capturing the inherent flexibility of proteins. Within a living cell, proteins continuously undergo fluctuations due to thermal motions, and the available crystal structures from the Protein Data Bank represent their local energetic minima. In line with this intuition, relevant studies seem to suggest that more fine-grained representations provide only minor improvements at best for a range of tasks (Wang et al., 2023). Meanwhile, residue-level representation has proven effective in some of the most prominent applications (Jumper et al., 2021; Watson et al., 2022). Although we consider the detailed atomic structure of interfacial side chains to be flexible, we assume the protein backbone to be relatively rigid (Townshend et al., 2019).

Formally, a protein-protein interface consisting of n residues is represented as a point cloud $\mathbf{X} \in \mathbb{R}^{n,3}$ of C_α atoms. The points are further associated with type-0 and type-1 features (e.g., invariant and equivariant under SE(3) transformations), denoted as $\mathbf{F}_0 \in \mathbb{R}^{n,21}$ and $\mathbf{F}_1 \in \mathbb{R}^{n,3}$. For the purpose of this study, the features are chosen to be minimalistic. Specifically, the type-0 features are set to:

$$\mathbf{f}_{0,i} = [partner(i), one\_hot(i)] \in \mathbb{R}^{21}, \tag{6.1}$$

where $partner(i)$ returns 0 or 1, depending on which of the two interacting chains the ith residue belongs to, and $one\_hot(i)$ returns the one-hot encoded amino acid type. The type-1 features capture the orientation of the corresponding side chains:

$$\mathbf{f}_{1,i} = C^*_\beta(i) - C_\alpha(i) \in \mathbb{R}^3. \tag{6.2}$$

The $C_\alpha(i)$ vector is represented by the alpha-carbon coordinates of the ith residue, and $C^*_\beta(i)$ corresponds to the coordinates of a virtual beta-carbon, calculated based on idealized geometry:

$$\mathbf{b} = C_\alpha - N, \tag{6.3}$$

$$\mathbf{c} = C - C_\alpha, \tag{6.4}$$

$$\mathbf{a} = \mathbf{b} \times \mathbf{c}, \tag{6.5}$$

$$C^*_\beta = -0.58273431\mathbf{a} + 0.56802827\mathbf{b} - 0.54067466\mathbf{c} + C_\alpha, \tag{6.6}$$

where C and N represent the other corresponding backbone atoms (Dauparas et al., 2022). Virtual beta-carbons provide an effective approximation of real ones and offer several advantages. First, they resolve the representation of glycine, the amino acid that lacks a beta-carbon. Second, virtual beta-carbons are beneficial for the proposed self-supervised learning schema discussed below.

An interfacial point cloud is additionally associated with an auxiliary undirected k-NN (k-nearest neighbors) graph $G = (V, E)$. In detail, $|V| = n$, where n is the number of residues in the interface, and two nodes $i, j \in V$ are considered adjacent if at least one of them is among the k nearest neighbors of the other one with respect to coordinates $\mathbf{X}$. In summary, each protein–protein interface is, therefore, characterized by residue positions $\mathbf{X}$, their associated features $\mathbf{F}_0, \mathbf{F}_1$, and a k-NN graph G.

6.1.2 Architecture

Representing a protein–protein interface as a point cloud allows utilizing the potential of modern geometric deep learning. Specifically, we employ SE(3)-Transformer proposed by Fuchs et al. (2020) as the foundation of our model. SE(3)-Transformer operates as a point cloud to point cloud encoder and, like most contemporary deep-learning architectures, is composed of L equivariant blocks (or layers). In the context of SE(3)-Transformer, the features associated with the points of a point cloud are referred to as fibers. Formally, a **fiber** is a set of all type-0, type-1, ..., type-l features associated with a point. One block of SE(3)-Transformer, therefore, maps fibers to new fibers while preserving SE(3)-equivariance. The versatility of the fiber-based representation is a significant advantage of SE(3)-Transformer compared to some other state-of-the-art SE(3)-equivariant approaches. For example, the EGNN by Satorras et al. (2021) can serve a similar purpose. However, it is not immediately apparent how to apply EGNN to equivariant learning from type-1 features rather than solely from the type-0 ones. Additionally, there exist other methods that go in line with SE(3)-Transformer, such as recently developed Equiformer (Liao and Smidt, 2022). Nevertheless, in this study, we proceed with the former approach due to its proven well-established applications (Watson et al., 2022).

As the name implies, SE(3)-Transformer is heavily influenced by the revolutionary Transformer architecture (Vaswani et al., 2017). In fact, SE(3)-Transformer

modifies the self-attention block from the original Transformer to achieve equivariance. The core concept behind constraining the Transformer block to be an SE(3)-equivariant function involves attaining equivariant key, query, and value matrices $\mathbf{K}, \mathbf{Q}, \mathbf{V}$. This can be accomplished by restricting the linear projections $\mathbf{W}_K, \mathbf{W}_Q, \mathbf{W}_V$ used in the Transformer to the combinations of equivariant basis functions, which consequently leads to equivariant self-attention. For a more comprehensive understanding, we recommend the well-written original paper by Fuchs et al. (2020).

Despite its name, SE(3)-Transformer is conceptualized and implemented as a graph neural network (see (Joshi, 2020) for the close relationship between the Transformer and graph neural networks). This means that, due to the constraints of equivariance, self-attention is, in fact, realized as a point-wise aggregation rather than a single series of large matrix multiplications updating all tokens at a time. Although this results in higher computational demands, it also offers the advantage of easy adaptation of self-attention to a graph structure (i.e., message passing). This can significantly reduce the quadratic computational complexity of self-attention and introduce a better inductive bias of locality by utilizing an appropriate graph structure.

Thus, to learn from a protein–protein point cloud specified by coordinates $\mathbf{X} \in \mathbb{R}^{n,3}$, type-0 features $\mathbf{F}_0 \in \mathbb{R}^{n,21}$, type-1 features $\mathbf{F}_1 \in \mathbb{R}^{n,3}$ and a k-NN graph G, we utilize a series of SE(3)-Transformer blocks $f^{(i)}$ with the goal of achieving the final invariant point embeddings $\mathbf{H} \in \mathbb{R}^{n,d_{out}}$:

$$\mathbf{H}_0^{(1)}, \mathbf{H}_1^{(1)}, \ldots, \mathbf{H}_{deg}^{(1)} = f^{(0)}(\mathbf{X}, G, \mathbf{F}_0, \mathbf{F}_1), \tag{6.7}$$

$$\mathbf{H}_0^{(i+1)}, \mathbf{H}_1^{(i+1)}, \ldots, \mathbf{H}_{deg}^{(i+1)} = f^{(i)}(\mathbf{X}, G, \mathbf{H}_0^{(i)}, \mathbf{H}_1^{(i)}, \ldots, \mathbf{H}_{deg}^{(i)}), \tag{6.8}$$

$$\mathbf{H} := \mathbf{H}_0^{(L)}, \tag{6.9}$$

where the dimensions and degrees of $f^{(i)}$ are set appropriately. A $\mathbf{H}_l^{(i)}$ matrix contains hidden type-l features given by the output of the ith SE(3)-Transformer block. Consequently, the *deg* value is a hyperparameter of the architecture defining the maximum hidden feature type (known as degree), as well as the d_{hidden} dimension specifying the common domain $\mathbb{R}^{d_{hidden}}$ for all hidden features in $\mathbf{H}_l^{(i)}$ for $1 < i < L$ and any l. We further refer to the encoder function given by the above equations shortly as the SE(3)-encoder f and due to the properties of SE(3)-Transformer it has the desired property of invariance to any rigid transformation

given by a rotation matrix $\mathbf{R} \in \mathbb{R}^{3,3}$ and a translation vector $\mathbf{t} \in \mathbb{R}^3$:

$$f : \mathbf{X}, G, \mathbf{F}_0, \mathbf{F}_1 \mapsto \mathbf{H}, \tag{6.10}$$

$$f(\mathbf{X}, G, \mathbf{F}_0, \mathbf{F}_1) = f(\mathbf{X}\mathbf{R} + \mathbf{1}\mathbf{t}^T, G, \mathbf{F}_0, \mathbf{F}_1), \tag{6.11}$$

where $\mathbf{1} \in \mathbb{R}^{n,1}$ is a matrix full of ones. In plain words, the encoding process consists of lifting the initial type-0 and type-1 features to a higher-dimensional and higher-degree space, "rethinking" their abstract representation by applying several other layers, and finally obtaining invariant high-dimensional per-point features $\mathbf{H}$ by extracting the type-0 output.

6.1.3 Training and inference

In this section, we describe the main motivation and advantage of PPIFORMER – leveraging a vast amount of annotated data, potentially encompassing all crystallized protein–protein interactions. As discussed in Chapter 2, the primary challenge in protein design lies in the non-additive combinatorial nature of mutations, which is nearly impossible to learn from limited data. Considering that the best available dataset of annotated interfacial mutations, SKEMPI2, contains only seven thousand entries, its potential for training rather than validation is questionable. The review by Geng et al. (2019b) and our case study on staphylokinase design highlight the unreliability of state-of-the-art methods fitted to SKEMPI2.

In this work, we propose to overcome the data scarcity constraint by generating millions of artificial training examples of a nature similar to the task of mutation effect prediction. We reason that training a model to complete missing amino-acid side chains in protein–protein interface structures is similar to predicting mutational effects, as both tasks require understanding the principles of amino acid substitutions. For this artificial task, we can generate a virtually unlimited number of training examples by masking random combinations of interfacial residues and training the model to classify them. Furthermore, we believe that masked modeling can give rise to generally-powerful neural representations of protein–protein interactions, as the approach has revolutionized the fields of natural language processing and computer vision, as well as protein sequence understanding (Balestriero et al., 2023; Meier et al., 2021).

Formally, we define masked modeling as the task of predicting the amino acid types (i.e. side chains) of residues with masked features. The first step to generate

a single training example involves sampling a random protein–protein interface and randomly selecting one of its partnering chains $p \in \{0, 1\}$. Then, we sample a set of indices to mask, $M \subset \{i \in \{1, \ldots, n\} \mid partner(i) = p\}$. This implies that we only mask amino acids from one partner at a time, which corresponds to a practical scenario of protein design where one protein is typically fixed (e.g. microplasmin) while the other one (e.g. staphylokinase) is being designed.

After sampling the set M, we mask the corresponding features. Specifically, we mask the type-0 features $\mathbf{F}_0$ to obtain $\mathbf{F}_0^*$, defined as follows:

$$\mathbf{f}_{0,i} = \begin{cases} [\mathbb{1}_{partner(i)=p}, \mathbf{0}] \text{ if } i \in M, \\ [\mathbb{1}_{partner(i)=p}, \mathbf{f}_{0,i}[2:]] \text{ if } i \notin M, \end{cases} \tag{6.12}$$

where $\mathbb{1}_{partner(i)=p}$ is an indicator that returns 0 or 1, ensuring training invariance to partner choice. In other words, we mask the one-hot amino acid types $\mathbf{f}_{0,i}[2:]$ of selected residues with zeroes and leave the rest unchanged, additionally swapping partner information to have the interpretation of "belongs to/does not belong to the same chains as the masked residues". It is important to note that we do not mask type-1 features, taking advantage of setting them to virtual beta-carbons. Using the unmasked real beta-carbons could result in data leakage, as they may contain amino acid-specific information. In contrast, virtual beta-carbons solely represent the backbone structure of a protein, remaining agnostic to specific side chains.

To train the model from masked representations, we define a simple classifier to predict amino acid types of masked residues:

$$g : \mathbf{H} \mapsto \mathbf{P}, \tag{6.13}$$

where $\mathbf{H}$ contains the final hidden features of the points and $\mathbf{P} \in \mathbb{R}^{n,20}$ is a probability matrix defining the predicted categorical distribution for each point in the corresponding rows. The classifier acts point-wise with $\mathbf{p}_i = softmax(g_h(\mathbf{h}_i))$, where $g_h : \mathbb{R}^{d_{out}} \to \mathbb{R}^{20}$ is a multi-layer perceptron and *softmax* is an ordinary softmax function such that $softmax_j(\mathbf{x}) = \frac{e^{x_j}}{\sum_{k=1}^{20} e^{x_k}}$.

Finally, training a model from one example includes sampling a random protein–protein interface given by $\mathbf{X}, G, \mathbf{F}_0, \mathbf{F}_1$, masking type-0 features $\mathbf{F}_0$ to obtain $\mathbf{F}_0^*$ and making a forward pass to obtain probabilities

$$\mathbf{P} = g(f(\mathbf{X}, G, \mathbf{F}_0^*, \mathbf{F}_1)), \tag{6.14}$$

where g is the classifier given by Equation (6.13) and g is the SE(3)-encoder given by Equation (6.12). All the parameters in the underlying SE(3)-Transformer blocks and the classifier are then updated with a gradient descent step minimizing the cross-entropy loss on masked amino acid types:

$$\mathcal{L} = -\sum_{i \in M} \sum_{j=1}^{20} \log\left(p_{ij}\right) \mathbb{1}_{j=wt(i)}, \tag{6.15}$$

where $wt(i)$ returns the index of a wild-type amino acid (i.e. class) of the ith residue and p_{ij} is the corresponding predicted probability of the wild type. The loss is thus minimized when the model correctly predicts all the masked wild-type amino acids given their structural neighborhoods. In practice, we train the model with mini-batches of masked interfaces and for each epoch we sample new masked indices M for each interface, which is known as dynamic masking (Liu et al., 2019).

We expect that the model trained in the described way, can be practically useful for a variety of downstream tasks. In this study, we concentrate on protein design applications and demonstrate how the PPIFORMER method can be employed to predict $\Delta\Delta G$. To estimate the effect of a single-point mutation, one can initially predict two probabilities corresponding to wild-type and mutated amino acids, followed by combining them with an appropriate binary function that captures their relative plausibility. The log-odds ratio heuristic, which is the difference between corresponding log-likelihoods, has been shown to accurately predict the effects of mutations across various domains (Riesselman et al., 2018). Therefore, we employ a well-established additive log-odds model to predict the effects of mutations in this work. Formally, given a multi-point mutation specified by the index set of mutated residues $M \subset \{1, \ldots, n\}$ (i.e. points in a point cloud representation) and a function $mut : M \to \{1, \ldots, 20\}$ returning the classes of mutations, we estimate the mutation effect as following:

$$\widehat{\Delta\Delta G} = \sum_{i \in M} \log(p_{i,wt(i)}) - \log(p_{i,mut(i)}). \tag{6.16}$$

Note that the underlying predicted probabilities $p_{i,j}$ in Equation (6.16) can be predicted in several meaningful ways (Meier et al., 2021). Namely, in this work we consider two possibilities. In one scenario one can mask the wild types and estimate $\Delta\Delta G$ solely from the context. In contrast, one can also estimate $\Delta\Delta G$

accounting for the wild types. Formally, in the first case the predictions can be obtained based on the masked type-0 features $\mathbf{F}_0^*$ and in the second one using the unmasked input $\mathbf{F}_0$. The former scenario better corresponds to the concept of masked training but requires performing one forward pass with different $\mathbf{F}_0^*$ for each unique M, rather than a single pass for any possible mutation on the same interface. Further, we refer to the probabilities predicted based on the masked features as **masked marginals**, while the wild-type-based predictions are referred to as **wild-type marginals**. It is important to note that in both scenarios the model does not require a mutant structure. It may offer several orders of magnitude faster $\Delta\Delta G$ prediction compared to state-of-the-art methods that rely on force-field simulations to build the mutated structure first.

6.2 Experimental setup

We evaluate the potential of our PPIFORMER using two performance measures. First, we assess the quality of self-supervised training by measuring the standard 20-class accuracy, i.e. the proportion of correctly classified masked amino acids. Additionally, we analyze the model's performance on the task of $\Delta\Delta G$ inference using zero-shot predictions, which means that we do not employ any further training and utilize PPIFORMER trained in a purely self-supervised way. We leave fine-tuning experiments for future work. To assess the model's performance on $\Delta\Delta G$ inference, we calculate Pearson and Spearman correlation coefficients between the predicted values and the values obtained in wet-lab experiments. The Pearson correlation measures linear dependency between values and reflects the correctness of the signs of predicted mutation effects, while the Spearman correlation measures the model's ability to order mutations correctly.

The most critical hyperparameter for training and performance analysis is the dataset choice. As described in Section 5.3.3, we consider four data partitions: training and validation parts of the large unannotated DIPS and training and validation folds of small $\Delta\Delta G$-annotated SKEMPI2. We observe that mining the DIPS dataset alone does not result in generalization to other data. Specifically, we find that achieving nearly 80% validation accuracy is easily possible when using the training and validation portions of DIPS for training and validation, respectively. However, subsequent accuracy measurement on the training part of SKEMPI2 yields poor accuracy below 10%, indicating strong overfitting. There-

fore, to measure the practical usability of the trained model, we use the entire DIPS dataset for training, and the training and validation parts of SKEMPI2 for validation and testing, respectively.

The other important hyperparameter of self-supervised learning via masked modeling is the masking regime, including the number of masked residues for each sampled interface. For the proof-of-concept purpose of this work, we simply mask one residue at a time (i.e. $|M| = 1$) with 100% probability, and leave the advanced training for future work. Accordingly, we only consider single-point mutations from SKEMPI2 for evaluation. To select the other hyperparameters of PPIFORMER for training, we partially explore the grid given by Table 6.1, evaluating 28 combinations in total.

We implement PPIFORMER in Python, primarily leveraging PyTorch, PyTorch Geometric, PyTorch Lightning, and Graphein (Paszke et al., 2019; Fey and Lenssen, 2019; Falcon and The PyTorch Lightning team, 2019; Jamasb et al., 2022). For training, we use the Czech supercomputer Karolina. For all hyperparameter configurations, we train a model for 24 hours on 8 NVIDIA A100 Tensor Core GPUs utilizing data parallelism. In 24 hours, a model can make up to 32,000 training steps, depending on the hyperparameter configuration.

6.3 Results

6.3.1 Ablations

We analyze the performance of all 28 selected instances of PPIFORMER with different hyperparameters and choose the best one according to manual inspection of validation accuracy and correlations. Interestingly, we find that setting the number of considered nearest neighbors k to 10 results in the highest validation accuracy, outperforming both $k = 5$ and $k = 20$. Although the combinations of the learning rate, batch size, weight decay, and other parameters that define model complexity do not have an additive effect on the model's performance, the optimal one is achieved with a learning rate of 10^{-3}, a batch size of 8 masked protein–protein interfaces per GPU, and disabling weight decay. The optimal combination of 7 layers and 8 heads is consistent with some of the best-established applications of the SE(3)-Transformer architecture (Fuchs et al., 2020; Watson et al., 2022). Overall, the best model contains 2,340,756 parameters. The dependency of the model's

Hyperparameter	Values
k neighbors	$\{5, \mathbf{10}, 20\}$
Learning rate	$\{10^{-4}, \mathbf{10^{-3}}, 20^{-3}, 50^{-3}, 10^{-2}\}$
Batch size per GPU	$\{2, 4, \mathbf{8}, 16\}$
Weight decay	$\{\mathbf{0}, 10^{-5}, 10^{-1}\}$
# SE(3)-Transformer layers L	$\{5, \mathbf{7}\}$
# SE(3)-Transformer heads	$\{2, 4, \mathbf{8}\}$
Hidden degree deg	$\{\mathbf{2}, 3\}$
Hidden dimension d_{hidden}	$\{4, \mathbf{32}\}$
# Classifier layers L	$\{\mathbf{1}, 2\}$
Output dimension d_{out}	$\{128, 256, \mathbf{512}\}$
$\Delta\Delta G$ inference kind	{`wt-marginals`, **`masked-marginals`**}

Table 6.1: Investigated hyperparameter space of PPIFORMER. The selected combination of hyperparameters is highlighted in bold.

$\Delta\Delta G$ scoring capabilities on the inference type is particularly interesting. We observe that the masked-marginals regime leads to consistently better performance. We will explore this phenomenon in future research. Below, we always assume masked marginals when discussing the results. For further evaluation, we select a model with the optimal hyperparameter configuration described above at training step 12095 (see Figure 6.2 A) and refer to it simply as PPIFORMER. This model achieves the validation accuracy of 0.2 and $\rho_{Pearson} = 0.25, \rho_{Spearman} = 0.28$.

6.3.2 PPIFORMER is capable of generalization under distribution shift

Training and validating machine-learning models on different parts of the same dataset may not accurately represent real-world performance, particularly when a natural schema for data partitioning is lacking. This is due to the potential influence of biases introduced during data preparation on validation performance. Consequently, in this study, we employ an extreme validation strategy by measuring generalization with respect to an independently collected dataset with a shifted distribution (Wiles et al., 2021), i.e. generalization from DIPS to SKEMPI2. Under such a stringent evaluation setup, the only way a model can achieve satisfactory performance is by effectively solving the task while minimizing reliance on potential biases.

Figure 6.2 A demonstrates that PPIFORMER is capable of robust generalization,

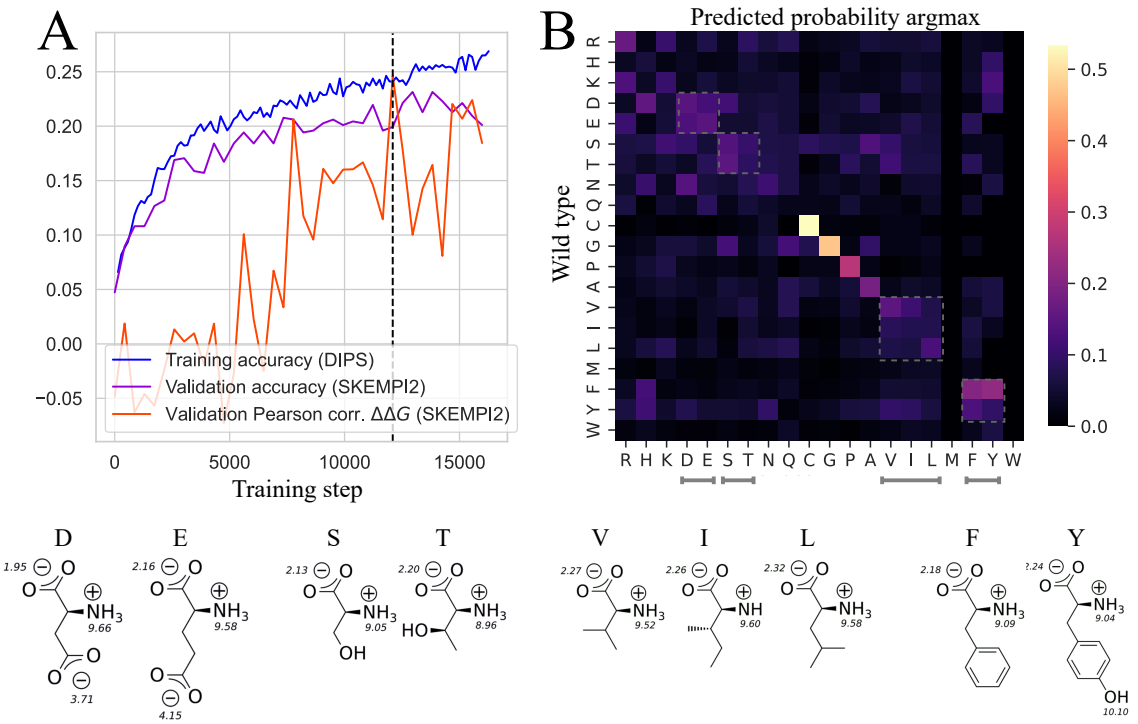

Figure 6.2: PPIFORMER generalizes through capturing biochemical principles. A) Learning curves of PPIFORMER with selected hyperparameters. The model was trained in a self-supervised way to predict missing amino acids on the whole DIPS dataset (blue curve). The purple and orange curves correspond to the validation on the training part of SKEMPI2, an independently-collected dataset with the shifted distribution. The orange curve illustrates the emerging capability of the model to score mutations, while not being explicitly optimized for the task. **B)** Confusion matrix for side-chain classification on the validation set, corresponding to the chosen training step 12095. Cysteine (C), glycine (G) and proline (P) are classified with high accuracy and are exactly three special cases of amino acids with unique biochemical properties[4] (see Figure 2.1 for the illustration of G and P). Molecular representations of amino acids corresponding to highlighted blocks (dashed squares) in the matrix are displayed below the figure. Aspartic (D) and glutamic (E) acids are the only negatively-charged amino acids and share a similar structure, resulting in their frequent mutual misclassification. Similar reasoning applies to other highlighted groups: all molecules within the groups share common properties including polarity/charge.

while Figure 6.2 B provides a rationale for this fact. The latter figure reveals that the model learns the statistical patterns that determine the contextual suitability of specific amino acids, capturing their biochemical properties. Special-case amino acids possess highly specific properties and are challenging to replace, which is reflected in their most accurate prediction by the model. In contrast, many amino acids have close analogs, which is captured by the model through their frequent mutual misclassification. Overall, given the chosen flexible residue-level data representation, misclassifications can be interpreted as potentially higher suitability of

the predicted amino acids compared to the wild-type ones. This implies that they may lead to increased binding affinity in the underlying interactions. Although we believe that the performance of PPIFORMER can be further improved (for instance, we hypothesize that achieving higher accuracy in classifying native amino acids may be advantageous), the presented results serve as a proof of concept for the strong generalization capabilities of the method.

6.3.3 PPIFORMER is capable of zero-shot transfer to mutation effect prediction

The conclusions related to misclassification in the previous paragraph suggest that PPIFORMER has the potential to identify favorable substitutions that increase binding affinity without any fine-tuning. Consequently, we assess the potential of zero-shot transfer of the model to predict the effects of mutations. Figure 6.2 A demonstrates the emergence of $\Delta\Delta G$ scoring capabilities during the training process. Further, we assess the performance of PPIFORMER on the test set (i.e. the validation part of SKEMPI2). Figure 6.3 shows a significant correlation between predictions and experimentally measured $\Delta\Delta G$ values (Pearson p-value $< 10^{-5}$). Since in a practical scenario, one is typically interested in scoring mutations on a single interface (e.g. staphylokinase–microplasmin), we additionally measure per-complex performance. Therefore, we calculate correlations on the subsets of the test set corresponding to individual complexes. We find that PPIFORMER achieves a high positive correlation for the majority of test interfaces.

Interestingly, we observe that the range of zero-shot predictions approximately aligns with the experimentally accepted range, often considered to be $[-8, 8]$ (Liu et al., 2021). We reason that this fact can be at least partially attributed to the log-likelihood nature of both the minimized cross-entropy loss and log-odds ratios employed for zero-shot predictions. This combination ensures an adequate range of predicted values, which may incidentally match the experimental one. On the other hand, the correspondence between the ranges may indicate an intriguing relationship between the emerging properties of predicted probabilities and the equilibrium constants of protein bindings. According to Equations (2.2), (2.4)

[4] https://en.wikipedia.org/wiki/Amino_acid#/media/File:ProteinogenicAminoAcids.svg

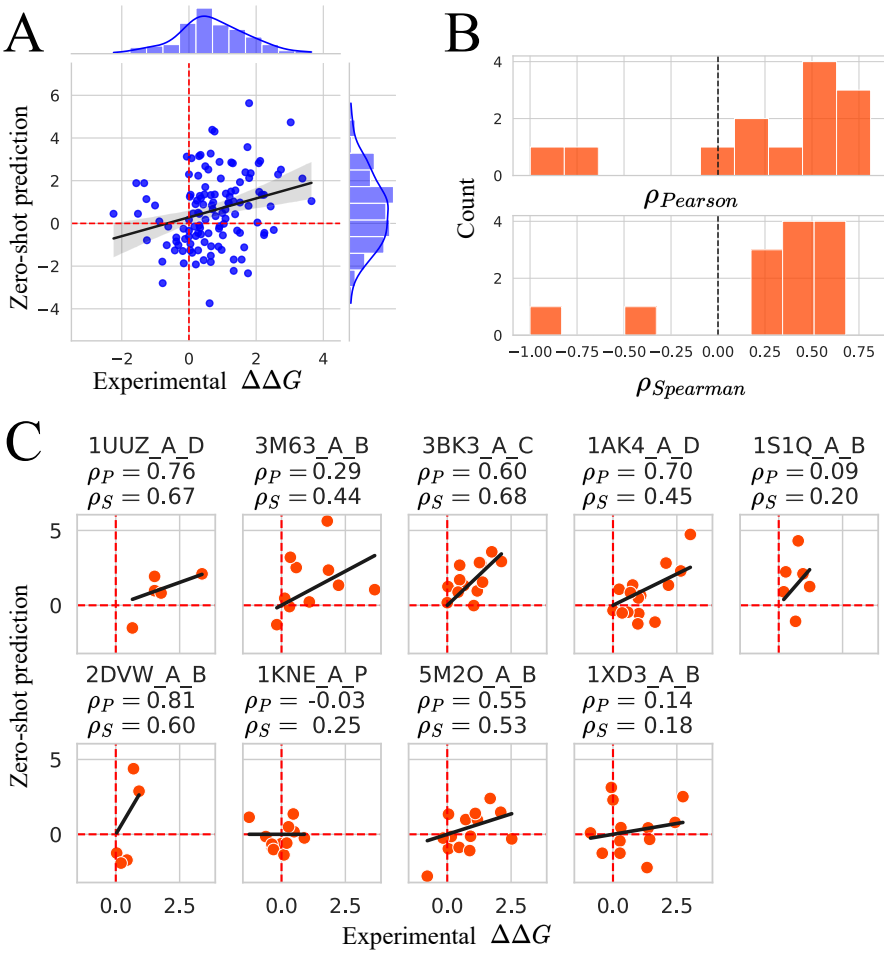

Figure 6.3: Zero-shot $\Delta\Delta G$ predictions by PPIFORMER correlate with experimental measurements. The figure visualizes the correlation of $\Delta\Delta G$ values from the SKEMPI2 test set and zero-shot predictions of PPIFORMER in the masked-marginals regime. **A)** Scatter plot corresponding to the whole test set ($\rho_{Pearson} = 0.27, \rho_{Spearman} = 0.29$). **B)** Distributions of the per-interaction correlations. The mean and median values are 0.25 and 0.47 for Pearson correlation, and 0.27 and 0.44 for Spearman correlation, respectively. **C)** Scatter plots corresponding to all test interactions with more than 4 annotated mutations.

and (6.16), for a single-point mutation of residue i it holds:

$$\Delta\Delta G = \Delta G_{mut} - \Delta G_{wt} = -RT(\ln(K_{wt}) - \ln(K_{mut})), \tag{6.17}$$

$$\widehat{\Delta\Delta G} = \log(p_{i,wt(i)}) - \log(p_{i,mut(i)}), \tag{6.18}$$

where $\Delta\Delta G$ and $\widehat{\Delta\Delta G}$ represent the predicted and experimental binding affinity changes, repsectively. Further, K_{wt} and K_{mut} are the equilibrium constants for the binding of wild-type and mutated complexes and $p_{i,wt(i)}$ and $p_{i,mut(i)}$ correspond to the predicted probabilities. Therefore, the match of the range of predictions with the experimental one, may, for example, suggest the potential of the zero-shot transfer of PPIFORMER for the inference of binding energy ΔG as well.

CHAPTER 7

Conclusion

In our study, motivated by the task of designing a next-generation thrombolytic staphylokinase, we have explored the problematics of machine learning for the design of protein–protein interactions. First, we applied several state-of-the-art methods to predict favorable mutations of staphylokinase with the potential of enhancing its thrombolytic activity. To accomplish the robust prediction, we have developed a consensus selection algorithm, which accounts for optimizing multiple protein properties while utilizing the collective knowledge of diverse methods. We have created a simple interactive website that visualizes the selection procedure. Several mutations proposed by the algorithm demonstrated high potential, and the ones approved by biochemistry experts are currently undergoing wet-lab validation at Loschmidt Laboratories to assess their influence on the thrombolytic activity of staphylokinase.

Our case study on staphylokinase revealed the strengths and weaknesses of the state of the art in machine learning for protein–protein interaction design. Namely, existing methods enable reliable preselection of plausible single-point substitutions. However, methods that could combine the preselected substitutions to construct multi-point mutations enhancing the binding affinity of the protein–protein interaction are severely missing. As the reliability of current methods stems from dependence on small annotated data, we propose to break this limitation by mining a vast amount of available crystallized protein–protein interactions. Therefore, in the first place, we have prepared and analyzed the big data of known protein–protein interactions from the whole Protein Data Bank. To achieve this, we have developed a fast algorithm for comparing protein–protein interfaces, en-

abling a large-scale analysis of all available interactions. Our analysis revealed strong, previously unaddressed biases of existent big protein–protein interaction data. Additionally, we identified strong limitations of the conventional utilization of such data, exemplified by the testing of machine learning models on data portions highly similar to the training data. Consequently, we have processed the extracted data to minimize biases, ensuring effective machine learning and fair evaluation.

Finally, we used the refined data to establish a novel self-supervised geometric deep learning model, PPIFORMER. The model leverages vast unannotated data by learning to solve an artificial task of predicting missing amino acids in the structures of protein–protein interactions. We demonstrated that the model effectively generalizes to independently-collected data with a different distribution by learning biochemical patterns. Furthermore, we showed that the proposed learning scheme enables PPIFORMER to predict the effects of mutations without any supervised training. This emergent property serves as a proof of concept for the approach, offering strong hope for overcoming the data scarcity issue that constrains existing methods for protein–protein interaction design.

In our future research, we will focus on unlocking the full potential of PPIFORMER. First, we will create a larger and more comprehensive database of crystallized protein–protein interactions for training, and enhance the architecture and training scheme of the model. We expect these improvements to result in a new, substantially more powerful version of PPIFORMER. Second, we plan to explore the fine-tuning potential of the model. While PPIFORMER has demonstrated its ability to score mutational effects without any supervision, we expect it to become a powerful protein-design assistant as a result of further fine-tuning. Additionally, we are intrigued to investigate the potential of the method for other tasks, such as predicting binding energy or scoring docking poses. Likewise, analyzing the neural representations of PPIFORMER may provide insights into complex biochemical phenomena such as epistasis or provide a state-of-the-art approach to comparing and clustering protein–protein interactions, which may, for example, enable the effective analysis of the vast human interactome. Finally, we will leverage PPIFORMER in the future rounds of staphylokinase design, and we expect the method to be broadly applicable in many case studies, including the design of other protein drugs.

Bibliography

B. Alberts, D. Bray, K. Hopkin, A. D. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Essential cell biology*. Garland Science, 2015.

A. Andreeva, D. Howorth, C. Chothia, E. Kulesha, and A. G. Murzin. Scop2 prototype: a new approach to protein structure mining. *Nucleic acids research*, 42 (D1):D310–D314, 2014. doi: 10.1093/nar/gkt1242.

R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum. A cookbook of self-supervised learning, 2023. URL https://doi.org/10.48550/arXiv.2304.12210.

M. Beckstette, R. Homann, R. Giegerich, and S. Kurtz. Fast index based algorithms and software for matching position specific scoring matrices. *BMC bioinformatics*, 7(1):1–25, 2006. doi: 10.1186/1471-2105-7-389.

H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic acids research*, 28 (1):235–242, 2000. doi: 10.1093/nar/28.1.235.

D. S. Biovia et al. Discovery studio modeling environment, 2017. URL https://www.3ds.com/products-services/biovia/products/molecular-modeling-simulation/biovia-discovery-studio/.

M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

I. Budowski-Tal, R. Kolodny, and Y. Mandel-Gutfreund. A novel geometry-based approach to infer protein interface similarity. *Scientific reports*, 8(1):8192, 2018. doi: 10.1038/s41598-018-26497-z.

P. V. Burra, Y. Zhang, A. Godzik, and B. Stec. Global distribution of conformational states derived from redundant models in the pdb points to non-uniqueness of the protein structure. *Proceedings of the National Academy of Sciences*, 106(26): 10505–10510, 2009. URL https://doi.org/10.1073/pnas.081215210.

T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785. URL http://doi.acm.org/10.1145/2939672.2939785.

S. Cheng, Y. Zhang, and C. L. Brooks. Pcalign: a method to quantify physicochemical similarity of protein-protein interfaces. *BMC bioinformatics*, 16(1):1–12, 2015. doi: 10.1186/s12859-015-0471-x.

G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*, 2022.

C. Dallago, J. Mou, K. E. Johnston, B. J. Wittmann, N. Bhattacharya, S. Goldman, A. Madani, and K. K. Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. *bioRxiv*, 2021. doi: 10.1101/2021.11.09.467890. URL https://www.biorxiv.org/content/early/2021/11/11/2021.11.09.467890.

R. Das and D. Baker. Macromolecular modeling with rosetta. *Annu. Rev. Biochem.*, 77:363–382, 2008. URL 10.1146/annurev.biochem.77.062906.171838.

J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. Wicky, A. Courbet, R. J. de Haas, N. Bethel, et al. Robust deep learning–based

protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022. doi: 10.1126/science.add2187.

Y. Dehouck, J. M. Kwasigroch, M. Rooman, and D. Gilis. Beatmusic: prediction of changes in protein–protein binding affinity on mutations. *Nucleic acids research*, 41(W1):W333–W339, 2013. doi: 10.1093/nar/gkt450.

J. Delgado, L. G. Radusky, D. Cianferoni, and L. Serrano. Foldx 5.0: working with rna, small molecules and a new graphical interface. *Bioinformatics*, 35(20): 4168–4169, 2019. doi: 10.1093/bioinformatics/btz184.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. doi: 10.1109/CVPR.2009.5206848.

E. J. Draizen, L. F. Murillo, J. Readey, C. Mura, and P. E. Bourne. Prop3d: A flexible, python-based platform for machine learning with protein structural properties and biophysical data. *bioRxiv*, pages 2022–12, 2022. URL https://doi.org/10.1101/2022.12.27.522071.

A. Elnaggar, H. Essam, W. Salah-Eldin, W. Moustafa, M. Elkerdawy, C. Rochereau, and B. Rost. Ankh: Optimized protein language model unlocks general-purpose modelling. *bioRxiv*, pages 2023–01, 2023. URL https://doi.org/10.48550/arXiv.2301.06568.

R. Evans, M. O'Neill, A. Pritzel, N. Antropova, A. Senior, T. Green, A. Žídek, R. Bates, S. Blackwell, J. Yim, et al. Protein complex prediction with alphafold-multimer. *BioRxiv*, pages 2021–10, 2021. URL https://doi.org/10.1101/2021.10.04.463034.

W. Falcon and The PyTorch Lightning team. PyTorch Lightning, Mar. 2019. URL https://github.com/Lightning-AI/lightning.

V. L. Feigin, M. Brainin, B. Norrving, S. Martins, R. L. Sacco, W. Hacke, M. Fisher, J. Pandian, and P. Lindsay. World stroke organization (wso): global stroke fact sheet 2022. *International Journal of Stroke*, 17(1):18–29, 2022. doi: 10.1177/17474930211065917.

M. Fey and J. E. Lenssen. Fast Graph Representation Learning with PyTorch Geometric, May 2019. URL https://github.com/pyg-team/pytorch_geometric.

D. M. Fowler and S. Fields. Deep mutational scanning: a new style of protein science. *Nature methods*, 11(8):801–807, 2014. URL https://doi.org/10.1038/nmeth.3027.

F. Fuchs, D. Worrall, V. Fischer, and M. Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *Advances in Neural Information Processing Systems*, 33:1970–1981, 2020. URL https://doi.org/10.48550/arXiv.2006.10503.

O.-E. Ganea, X. Huang, C. Bunne, Y. Bian, R. Barzilay, T. Jaakkola, and A. Krause. Independent se (3)-equivariant models for end-to-end rigid protein docking. *arXiv preprint arXiv:2111.07786*, 2021.

M. Gao and J. Skolnick. ialign: a method for the structural comparison of protein–protein interfaces. *Bioinformatics*, 26(18):2259–2265, 2010a. URL https://doi.org/10.1093/bioinformatics/btq404.

M. Gao and J. Skolnick. Structural space of protein–protein interfaces is degenerate, close to complete, and highly connected. *Proceedings of the National Academy of Sciences*, 107(52):22517–22522, 2010b. URL https://doi.org/10.1073/pnas.101282010.

Z. Gao, C. Jiang, J. Zhang, X. Jiang, L. Li, P. Zhao, H. Yang, Y. Huang, and J. Li. Hierarchical graph learning for protein–protein interaction. *Nature Communications*, 14(1):1093, 2023. URL https://doi.org/10.1038/s41467-023-36736-1.

C. Geng, A. Vangone, G. E. Folkers, L. C. Xue, and A. M. Bonvin. isee: Interface structure, evolution, and energy-based machine learning predictor of binding affinity changes upon mutations. *Proteins: Structure, Function, and Bioinformatics*, 87(2):110–119, 2019a. doi: 10.1002/prot.25630.

C. Geng, L. C. Xue, J. Roel-Touris, and A. M. Bonvin. Finding the $\delta\delta$g spot: are predictors of binding affinity changes upon mutations in protein–protein interactions ready for it? *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 9(5):e1410, 2019b. doi: 10.1002/wcms.1410.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

I. Hardcastle. 5.06 - protein–protein interaction inhibitors in cancer. In S. Chackalamannil, D. Rotella, and S. E. Ward, editors, *Comprehensive Medicinal Chem-*

istry III, pages 154–201. Elsevier, Oxford, 2017. ISBN 978-0-12-803201-5. doi: https://doi.org/10.1016/B978-0-12-409547-2.12392-3. URL https://www.sciencedirect.com/science/article/pii/B9780124095472123923.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991. URL https://doi.org/10.1016/0893-6080(91)90009-T.

W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020. URL https://doi.org/10.48550/arXiv.2005.00687.

A. R. Jamasb, R. V. Torné, E. J. Ma, Y. Du, C. Harris, K. Huang, D. Hall, P. Lio, and T. L. Blundell. Graphein - a python library for geometric deep learning and network analysis on biomolecular structures and interaction networks. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=9xRZlV6GfOX.

J. Jankauskaitė, B. Jiménez-García, J. Dapkūnas, J. Fernández-Recio, and I. H. Moal. Skempi 2.0: an updated benchmark of changes in protein–protein binding energy, kinetics and thermodynamics upon mutation. *Bioinformatics*, 35(3): 462–469, 2019. doi: https://doi.org/10.1093/bioinformatics/bty635.

Y. Jiang, L. Quan, K. Li, Y. Li, Y. Zhou, T. Wu, and Q. Lyu. Dgcddg: Deep graph convolution for predicting protein-protein binding affinity changes upon mutations. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2023. doi: 10.1109/TCBB.2022.3233627.

W. Jin, S. Sarkizova, X. Chen, N. Hacohen, and C. Uhler. Unsupervised protein-ligand binding energy prediction via neural euler's rotation equation. *arXiv preprint arXiv:2301.10814*, 2023.

C. Joshi. Transformers are graph neural networks. *The Gradient*, 12, 2020. URL https://thegradient.pub/transformers-are-graph-neural-networks/.

J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly accurate protein

structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. URL https://doi.org/10.1038/s41586-021-03819-2.

P. L. Kastritis and A. M. Bonvin. On the binding affinity of macromolecular interactions: daring to ask why proteins interact. *Journal of The Royal Society Interface*, 10(79):20120835, 2013. URL https://doi.org/10.1098/rsif.2012.0835.

O. Keskin, A. Gursoy, B. Ma, and R. Nussinov. Principles of protein- protein interactions: what are the preferred ways for proteins to interact? *Chemical reviews*, 108(4):1225–1244, 2008. URL https://doi.org/10.1021/cr040409x.

M. A. Ketata, C. Laue, R. Mammadov, H. Stärk, M. Wu, G. Corso, C. Marquet, R. Barzilay, and T. S. Jaakkola. Diffdock-pp: Rigid protein-protein docking with diffusion models. *arXiv preprint arXiv:2304.03889*, 2023.

Y. Laroche, S. Heymans, S. Capaert, F. De Cock, E. Demarsin, and D. Collen. Recombinant staphylokinase variants with reduced antigenicity due to elimination of b-lymphocyte epitopes. *Blood, The Journal of the American Society of Hematology*, 96(4):1425–1432, 2000. URL https://doi.org/10.1182/blood.V96.4.1425.

Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Y. LeCun and I. Misra, Mar 2021. URL https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/.

E. D. Levy. A simple definition of structural regions in proteins and its use in analyzing interface evolution. *Journal of molecular biology*, 403(4):660–670, 2010. doi: 10.1016/j.jmb.2010.09.028.

G. Li, S. Pahari, A. K. Murthy, S. Liang, R. Fragoza, H. Yu, and E. Alexov. Saambe-seq: a sequence-based method for predicting mutation effect on protein–protein binding affinity. *Bioinformatics*, 37(7):992–999, 2021. doi: 10.1093/bioinformatics/btaa761.

Y. Li, M. A. Rezaei, C. Li, and X. Li. Deepatom: A framework for protein-ligand binding affinity prediction. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 303–310. IEEE, 2019. URL https://doi.org/10.48550/arXiv.1912.00318.

Y.-L. Liao and T. Smidt. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. *arXiv preprint arXiv:2206.11990*, 2022. URL https://doi.org/10.48550/arXiv.2206.11990.

Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, A. dos Santos Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv*, 2022. URL https://doi.org/10.1101/2022.07.20.500902.

X. Liu, Y. Luo, P. Li, S. Song, and J. Peng. Deep geometric representations for modeling effects of mutations on protein-protein binding affinity. *PLoS computational biology*, 17(8):e1009284, 2021. URL https://doi.org/10.1371/journal.pcbi.1009284.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang. The expressive power of neural networks: A view from the width. *Advances in neural information processing systems*, 30, 2017. URL https://doi.org/10.48550/arXiv.1709.02540.

S. M. Marques, P. Kouba, A. Legrand, J. Sedlar, L. Disson, J. Planas-Iglesias, Z. Sanusi, A. Kunka, J. Damborsky, T. Pajdla, et al. Effects of alzheimer's disease drug candidates on disordered aβ42 dissected by comparative markov state analysis (covampnet). *bioRxiv*, pages 2023–01, 2023. URL https://doi.org/10.1101/2023.01.06.523007.

J. Meier, R. Rao, R. Verkuil, J. Liu, T. Sercu, and A. Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in Neural Information Processing Systems*, 34:29287–29303, 2021. URL https://doi.org/10.1101/2021.07.09.450648.

C. Mirabello and B. Wallner. Topology independent structural matching discovers novel templates for protein interfaces. *Bioinformatics*, 34(17):i787–i794, 2018. doi: 10.1093/bioinformatics/bty587.

C. M. Miton and N. Tokuriki. How mutational epistasis impairs predictability in protein evolution and design. *Protein Science*, 25(7):1260–1272, 2016. doi: 10.1002/pro.2876.

A. Morehead, C. Chen, A. Sedova, and J. Cheng. Dips-plus: The enhanced database of interacting protein structures for interface prediction. *arXiv preprint arXiv:2106.04362*, 2021.

R. Netzer, D. Listov, R. Lipsh, O. Dym, S. Albeck, O. Knop, C. Kleanthous, and S. J. Fleishman. Ultrahigh specificity in a network of computationally designed protein-interaction pairs. *Nature communications*, 9(1):5286, 2018. URL https://doi.org/10.1038/s41467-018-07722-9.

D. Nikitin, J. Mican, M. Toul, D. Bednar, M. Peskova, P. Kittova, S. Thalerova, J. Vitecek, J. Damborsky, R. Mikulik, et al. Computer-aided engineering of staphylokinase toward enhanced affinity and selectivity for plasmin. *Computational and structural biotechnology journal*, 20:1366–1377, 2022. URL https://doi.org/10.1016/j.csbj.2022.03.004.

S. Pahari, G. Li, A. K. Murthy, S. Liang, R. Fragoza, H. Yu, and E. Alexov. Saambe-3d: predicting effect of mutations on protein–protein interactions. *International journal of molecular sciences*, 21(7):2563, 2020. doi: 10.3390/ijms21072563.

S. Park and C. Seok. Galaxywater-cnn: prediction of water positions on the protein structure by a 3d-convolutional neural network. *Journal of Chemical Information and Modeling*, 62(13):3157–3168, 2022. URL https://doi.org/10.1021/acs.jcim.2c00306.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

J. Ribeiro, C. Ríos-Vera, F. Melo, and A. Schüller. Calculation of accurate interatomic contact surface areas for the quantitative analysis of non-bonded molecular interactions. *Bioinformatics*, 35(18):3499–3501, 2019. URL https://doi.org/10.1093/bioinformatics/btz062.

A. J. Riesselman, J. B. Ingraham, and D. S. Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature methods*, 15(10):816–822, 2018. URL https://doi.org/10.1038/s41592-018-0138-4.

C. H. Rodrigues, D. E. Pires, and D. B. Ascher. mmcsm-ppi: predicting the effects of multiple point mutations on protein–protein interactions. *Nucleic acids research*, 49(W1):W417–W424, 2021. doi: https://doi.org/10.1093/nar/gkab273.

V. G. Satorras, E. Hoogeboom, and M. Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021. URL https://doi.org/10.48550/arXiv.2102.09844.

J. Schymkowitz, J. Borg, F. Stricher, R. Nys, F. Rousseau, and L. Serrano. The foldx web server: an online force field. *Nucleic acids research*, 33(suppl_2): W382–W388, 2005. doi: 10.1093/nar/gki387.

E. Sevgen, J. Moller, A. Lange, J. Parker, S. Quigley, J. Mayer, P. Srivastava, S. Gayatri, D. Hosfield, M. Korshunova, et al. Prot-vae: Protein transformer variational autoencoder for functional protein design. *bioRxiv*, pages 2023–01, 2023. URL https://doi.org/10.1101/2023.01.23.525232.

C. Shen, J. Ding, Z. Wang, D. Cao, X. Ding, and T. Hou. From machine learning to deep learning: Advances in scoring functions for protein–ligand docking. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 10(1):e1429, 2020. URL https://doi.org/10.1002/wcms.1429.

W.-H. Shin, K. Kumazawa, K. Imai, T. Hirokawa, and D. Kihara. Quantitative comparison of protein-protein interaction interface using physicochemical feature-based descriptors of surface patches. *Frontiers in Molecular Biosciences*, 10, 2023. URL https://doi.org/10.3389/fmolb.2023.1110567.

R. Shroff, A. W. Cole, D. J. Diaz, B. R. Morrow, I. Donnell, A. Annapareddy, J. Gollihar, A. D. Ellington, and R. Thyer. Discovery of novel gain-of-function mutations guided by structure-based deep learning. *ACS synthetic biology*, 9(11):2927–2935, 2020. URL https://doi.org/10.1021/acssynbio.0c00345.

J. Skolnick, H. Zhou, and M. Brylinski. Further evidence for the likely completeness of the library of solved single domain protein structures. *The journal of physical chemistry B*, 116(23):6654–6664, 2012. URL https://doi.org/10.1021/jp211052j.

H. Stärk, O. Ganea, L. Pattanaik, R. Barzilay, and T. Jaakkola. Equibind: Geometric deep learning for drug binding structure prediction. In *International Conference*

on Machine Learning, pages 20503–20521. PMLR, 2022. URL https://doi.org/10.48550/arXiv.2202.05146.

M. Steinegger and J. Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017. URL https://doi.org/10.1038/nbt.3988.

M. Steinegger and J. Söding. Clustering huge protein sequence sets in linear time. *Nature communications*, 9(1):2542, 2018. URL https://doi.org/10.1038/s41467-018-04964-5.

L. Sumbalova, J. Stourac, T. Martinek, D. Bednar, and J. Damborsky. Hotspot wizard 3.0: web server for automated design of mutations and smart libraries based on sequence input information. *Nucleic acids research*, 46(W1):W356–W362, 2018. doi: 10.1093/nar/gky417.

K. Thyagharajan and G. Kalaiarasi. A review on near-duplicate detection of images using computer vision techniques. *Archives of Computational Methods in Engineering*, 28:897–916, 2021. URL https://doi.org/10.48550/arXiv.2009.03224.

M. Toul, D. Nikitin, M. Marek, J. Damborsky, and Z. Prokop. Extended mechanism of the plasminogen activator staphylokinase revealed by global kinetic analysis: 1000-fold higher catalytic activity than that of clinically used alteplase. *ACS Catalysis*, 12(7):3807–3814, 2022. URL https://doi.org/10.1021/acscatal.1c05042.

R. Townshend, R. Bedi, P. Suriana, and R. Dror. End-to-end learning on 3d protein structure for interface prediction. *Advances in Neural Information Processing Systems*, 32, 2019. URL https://doi.org/10.48550/arXiv.1807.01297.

M. van Kempen, S. S. Kim, C. Tumescheit, M. Mirdita, C. L. Gilchrist, J. Söding, and M. Steinegger. Foldseek: fast and accurate protein structure search. *Biorxiv*, pages 2022–02, 2022. URL https://doi.org/10.1101/2022.02.07.479398.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. doi: https://doi.org/10.48550/arXiv.1706.03762.

L. Wang, H. Liu, Y. Liu, J. Kurtin, and S. Ji. Learning hierarchical protein representations via complete 3d graph networks, 2023.

M. Wang, Z. Cang, and G.-W. Wei. A topology-based network tree for the prediction of protein–protein binding affinity changes following mutation. *Nature Machine Intelligence*, 2(2):116–123, 2020. doi: https://doi.org/10.1038/s42256-020-0149-6.

J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, et al. Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models. *bioRxiv*, pages 2022–12, 2022. URL https://doi.org/10.1101/2022.12.09.519842.

O. Wiles, S. Gowal, F. Stimberg, S. Alvise-Rebuffi, I. Ktena, K. Dvijotham, and T. Cemgil. A fine-grained analysis on distribution shift. *arXiv preprint arXiv:2110.11328*, 2021.

K. E. Wu, K. K. Yang, R. v. d. Berg, J. Y. Zou, A. X. Lu, and A. P. Amini. Protein structure generation via folding diffusion. *arXiv preprint arXiv:2209.15611*, 2022a.

R. Wu, F. Ding, R. Wang, R. Shen, X. Zhang, S. Luo, C. Su, Z. Wu, Q. Xie, B. Berger, et al. High-resolution de novo structure prediction from primary sequence. *BioRxiv*, pages 2022–07, 2022b. URL https://doi.org/10.1101/2022.07.21.500999.

P. Xiong, C. Zhang, W. Zheng, and Y. Zhang. Bindprofx: assessing mutation-induced binding affinity change by protein interface profiles with pseudo-counts. *Journal of molecular biology*, 429(3):426–434, 2017. doi: 10.1016/j.jmb.2016.11.022.

Y. Zhang and J. Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57 (4):702–710, 2004. doi: 10.1002/prot.20264.

G. Zhou, M. Chen, C. J. Ju, Z. Wang, J.-Y. Jiang, and W. Wang. Mutation effect estimation on protein–protein interactions using deep contextualized representation learning. *NAR genomics and bioinformatics*, 2(2):lqaa015, 2020a. URL https://doi.org/10.1093/nargab/lqaa015.

J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *AI open*, 1: 57–81, 2020b. URL https://doi.org/10.48550/arXiv.1812.08434.

APPENDIX A

Acronyms

DIPS Database of Interacting Protein Structures

PDB Protein Data Bank

PPI Protein–protein interaction

SAK Staphylokinase

SKEMPI2 Structural database of Kinetics and Energetics of Mutant Protein Interactions v2.0

Appendix B

Contents of enclosed CD

- `cd` . . . . . Enclosed CD
  - `sak` . . . . . Python package containing experiments from Chapter 4
  - `ppi` . Python package containing experiments from Chapter 5 and Chapter 6
  - `mutils` . . . . . Supplementary Python package for experiments
  - `tex` . . . . . Directory of LaTeX source codes of the thesis
  - `thesis.pdf` . . . . . Text of the thesis in PDF format