Learning the language of biomolecular interactions

by

Samuel Sledzieski

B.S., University of Connecticut (2019)
S.M., Massachusetts Institute of Technology (2021)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by: Samuel Sledzieski
Department of Electrical Engineering and Computer Science
May 17, 2024

Certified by: Bonnie Berger
Simons Professor of Mathematics
Thesis Supervisor

Accepted by: Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Learning the language of biomolecular interactions

by

Samuel Sledzieski

Submitted to the Department of Electrical Engineering and Computer Science
on May 17, 2024 in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILSOPHY

ABSTRACT

Proteins are the primary functional unit of the cell, and their interactions drive cellular function. Interactions between proteins are responsible for a wide variety of functions raning from catalytic activity to cellular transport and signaling, and interactions between small molecules and proteins are the foundation of many therapeutics. However, the experimental determination of these interactions is expensive and relatively slow, limiting the ability to model interactions at genome scale. It is therefore critical to develop computational approaches for modeling these interactions. Unsupervised language models trained on amino acid sequences, namely protein language models, learn patterns in sequence evolution that encode protein structure and function. These protein language models are thus a powerful tool for extracting features of proteins, enabling the adoption of lightweight downstream models. Here, we present novel machine learning techniques for adapting protein language modeling to the prediction of protein interactions at scale, enabling de novo interaction network inference and large-scale drug compound screening. We show that these methods achieve state-of-the-art performance, and allow us to discover new biology and therapeutic candidates. In addition, we introduce methods for efficient training and adaptation of these models, and outline several applications which take advantage of the scale enabled by lightweight models. As a whole, this thesis demonstrates how computational advances in language modeling and the massive growth of data brought about by the sequencing revolution can be leveraged to tackle the genotype-to-phenotype challenge in biology, and lays the groundwork for more widespread adoption of these techniques for proteomic modeling.

Thesis supervisor: Bonnie Berger
Title: Simons Professor of Mathematics

To my parents, Ron and Nana. None of this is possible without you.

Acknowledgments

It seems almost impossible to put into a few pages the full impact of every contribution that makes a PhD possible; what follows is my attempt to make sense of the community that has supported me as I've pursued this opportunity, and to acknowledge the profound impact that they've had on this work and on me. My gratitude is first and foremost to my advisor, Bonnie Berger—Bonnie has an unparalleled vision for science, is uncompromising in her support, and has fostered an environment where I can explore, learn, fail, and grow, for which I am forever thankful. It is equally difficult to overstate my thanks for Lenore Cowen and Rohit Singh, who have powered me through every week of the last five years with their guidance and belief in me, and from whom I have learned how to be a better scientist.

I've been extremely fortunate to have a long list of supportive mentors prior to and through my PhD. Thanks to Manolis Kellis and Collin Stultz, not only for their place on my committee, but for their constant support of me since the first weeks of graduate school as research and academic advisors. Thanks especially to Mukul Bansal, who served as my undergraduate advisor, for introducing me to the world of computation and biology, taking a chance on me when I was only beginning to understand both, and preparing me for a lifetime of research. Thanks also to Ion Mandoiu, Paul Lewis, Jonah Tower, Jon Bloom, Meghana Kshirsagar, and Francois Seneca. Their fingerprints are likewise on this thesis, as I had the pleasure to grow and learn from them at each stop on the way.

My graduate experience has been extraordinarily enriched by the strength of the community at MIT; I'm especially grateful for the opportunity to work alongside and learn from the

members of the Berger Lab that I've overlapped with during my time here. Thanks to the postdocs and fellow graduate students in the Berger Lab—Ariya Shajii, Tristan Bepler, Brian Hie, Ellen Zhong, Ashwin Narayan, Ben DeMeo, Sarah Nyquist, YounHun Kim, Max Sherman, Adam Yaari, Alex Wu, David Froelicher, Ruochi Zhang, Bariş Ekim, Bowen Jing, Lillian Zhang, Shuvom Sadhuka, Matt Hong, Anna Sappington, Daniel Lazarev, Ben Fefferman, Lena Erlach, Daniel Edelman, Seokmin Ha, Daniel Shaffer, Megan Le, and Onkur Gujral—as well as to those that I've had the opportunity to serve as a mentor for—Jesse Yang, Lynn Tao, Bhushan Mohanraj, Ji Won Kim, and Adrina Tang. Thanks especially to the administrative assistants in the Berger Lab—Patrice Macaluso, Jeff Taft, and Pepe Abola, whose kindness, knowledge and assistance has made my life orders of magnitude easier. Thanks also to Alica Duarte, Janet Fischer, Leslie Kolodziejski, Meredith Bittrich, Liza Ruano, and Anthony Vasquez in the EECS graduate office for their help in navigating every step of the PhD process.

I've likewise had the pleasure of working with a number of other collaborators and co-authors outside of the Berger Lab. My thanks goes especially to Bryan Bryson, Minkyung Baek, Rahul Dodhia, and Juan Lavista Ferres who all contributed to work which appeared in this thesis. Special thanks to Kapil Devkota, a consistent collaborator and friend who has also contributed to work in this thesis. Thanks also to the other members of the Cowen Lab—Mert Erden, Monsurat Olaosebikan, Charlotte Versavel, Polina Shpilker, and Faith Ocitti—as well as to the broader community of scientists—Judith Klein-Seetharaman, Hollie Putnam, Noah Daniels, Natassja Lewinski, YK Yang, Lokender Kumar, Nathanael Brenner, Sumaira Zaman, and Jessica Wu—I've had the opportunity to work and publish with, and to learn from. Each person has brought a unique perspective and set of knowledge, as well as camaraderie, which has greatly impacted my growth as a scientist and my experience in graduate school.

The time I've been able to dedicate to my PhD is made possible only by the quality of life and happiness I've been fortunate enough to have outside of it. Thank you to all of my

friends that contributed in one way or another to this happiness—whether I've known you since graduate school or elementary school, whether I've lived with you or had to fly across the country to see you. Thanks to Daniel Boba, Emmett Juliard, Dereck Li, Kevin Matthews, Ricky Ribas, Nathaniel Bell, Kenny Bisch, Mitchell Cascella, Tim Cohn, Anthony Desiato, CJ Legato, Justin Lin, Jack O'Neill, Frank Sinapi, Garrett Collins, Anthony D'Andrea, Killian Green, Daniel MacLean, Samantha Morales, Will Reid, Matthew Bonfitto, Zeke Delilah, Owen Höglund, and Max Wade for the love and support you've shown me.

Finally, thank you to my partner, Demmy. I am so, so grateful for every single day that I get to spend with you, and each day makes me more excited for the next. Thank you to my sisters, Sydney and Summer. You both inspire me, and I'm so proud of the lives you've each grown into. I can't write anything here that you all don't already know, but I love you, and I'll never let you forget it.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Molecular interactions tell the story of the cell

The protein sits at the heart of complex life. The RNA world hypothesis [1] suggest that early life started with RNA performing the dual role of both information storage and catalyst. As life developed and advanced in complexity, these roles were diversified—DNA evolving as the primary information storage, and proteins as the primary functional unit. If DNA is the code, proteins are the compiled software. If DNA is the blueprint, proteins are the machines, manifest and functional. If DNA is the script, proteins are the actors, embodying the characters and bringing them to life. Proteins play a role in nearly every cellular function, including the immune response [2], [3], catalytic activity [4], transport [5], structure [6], signaling [7], and gene regulation [8]. Thus, to understand life it is fundamental to study the functions of proteins, and the relationship between their sequence and function.

However, studying proteins (or any biomolecule) on their own will never fully capture the complexity of life, because this complexity arises through their *interactions*. The story of any complex system is told through the network of interactions between constituent parts. Servers send packets to each other to form the internet. Machines pass material to one another to form assembly lines and factory floors. Actors exchange lines and the story unfolds.

The focus of this thesis is proteins, and the way that their interactions with each other and with other biomolecules tell the story of cellular function. Specifically, we explore **how the interaction network of a protein is encoded in its sequence**.

Our interest in interaction networks goes beyond just *understanding* cellular function; we also seek to explore ways of *modulating* this function. Many disease phylogenies can be traced to the dysfunction of a protein, or the over-activity thereof. Therapeutics such as small-molecule drugs function likewise through interactions—by binding to the correct target, the dysfunctional protein activity can be regulated. Small molecule-drugs are stage directions, external instructions that can change how the script is read out, how the actors engage with one another, that can remove one from the scene altogether. Thus, we expand our scope to consider not only interactions between proteins, but the interactions of other biomolecules with proteins, and we explore **how to develop effective small-molecule therapeutics against a given target.**

The relating of the genetic code to a language, a script that is read out by proteins, is more than just convenient analogy. In fact, the sequences of amino acids which make up a protein sequence share many properties with natural language. They are made up of an characters combined into words and then larger sentences, such as the conserved domains that recur in evolutionarily related proteins. And like a language, context matters. Just as the word "keyboard" refers to different concepts when preceded by "computer" or "piano", the same amino acid or motif will have a different function depending on the surrounding protein. A major theme of this thesis is that the computational tools developed to model language are often equally suitable when applied to protein sequences, and that these tools can efficiently compress the complexity of sequence variation into vector representations which are amenable to machine learning. Then, we also explore **how computational can advances in language modeling allow us to read the language of biomolecular interactions** and infer the scenes that play out?

1.2 A note on the value of scalable modeling in biology

One of the challenges of accurately modeling biology is the scale of the data and its possible combinations. Throughout this thesis, we place particular emphasis on the efficiency of our algorithms. The human proteome contains approximately 20,000 unique proteins, and that number balloons when you expand beyond humans, including proteins in model organisms, the microbiome, viruses, and beyond. The space of small molecules is potentially even larger—at the time of writing, there are approximately 500,000 drugs in DrugBank [9], and over 20 million compounds in the ZINC database [10].

The pairwise nature of interaction means that as the number of individual units grows, the number of possible interactions grows quadratically. The complexity of living systems arises due to network effects; it is therefore critical that we are able to model these effects at the scale at which they occur. We intentionally design algorithms here that are able to efficiently predict at the required scale. The efficient transfer learning that protein language models enable allows us to design lightweight, data-efficient models that can predict genome-wide networks, infer functional pathways, search massive compound libraries, and perform deep *in silico* mutational scans. High-throughput computational methods will likely enable advances in personalized medicine, where inference is needed for the unique genotype, including somatic mutations, of each individual—we thus believe that it is critical to keep an eye towards the efficiency and accessibility of computational methods.

1.3 Thesis organization

We begin this thesis in Chapter 2 with an overview of the relevant background which will be helpful in understanding the contributions of this work. We cover several of the core biological and computational concepts that underlie the work presented herein, and while not providing complete details, we direct the reader to external references for further reading on

these topics. This thesis is designed to be accessible to an advanced undergraduate, perhaps with background in computation or biology but not necessarily both, and Chapter 2 offers a primer to allow such a reader to understand this thesis and begin work in these and related areas.

Chapters 3-6 comprise the main body of this work. Each chapter starts with an overview, contextualizing the chapter in the context of the thesis as a whole, followed by a more detailed introduction to the chapter contents, a description of the algorithmic advances, and several validation and case studies. We end each chapter with a short perspective on limitations and future directions. Each chapter can be read somewhat independently, although Chapter 4 leans heavily on material introduced in Chapter 3.

In Chapters 3 and 4, we explore the problem of predicting protein-protein interactions (PPI) under limited information domains. In Chapter 3, we introduce the use of protein language models (PLMs) for PPI and propose a structurally-driven method for modeling sequence-only interaction. We show that this approach drastically improves the ability to predict PPI in a wide variety of under-studied species. This chapter is adapted from work which originally appeared in RECOMB 2021 and was published in:

- **Sledzieski, S.***, Singh, R.*, Cowen, L., & Berger, B. (2021). D-SCRIPT translates genome to phenome with sequence-based, structure-aware, genome-scale predictions of protein-protein interactions. Cell Systems, 12, 1–14. [11]

In Chapter 4, we expand the scope to include two orthogonal sources of information which may be available when modeling PPI. We develop integrative frameworks which adapt to data availability, including network topology information using algorithms from graph theory and protein 3D structure information using sequence quantization techniques. This chapter acts as a companion to the previous, demonstrating how the framework introduced in Chapter 3 can be expanded upon and improved. This chapter is adapted from work which originally appeared at ISMB 2022 and 2023 and was published in:

- Singh, R.*, Devkota, K.*, **Sledzieski, S.**, Berger, B., & Cowen, L. (2022). Topsy-Turvy: Integrating a global view into sequence-based PPI prediction. Bioinformatics, 38(Supplement_1), i264–i272. [12]

- **Sledzieski, S.***, Devkota, K.*, Singh, R., Cowen, L., & Berger, B. (2023). TT3D: Leveraging pre-computed protein 3D sequence models to predict protein-protein interactions. Bioinformatics, btad663. [13]

While protein-protein interactions are critical to cellular function, many of the therapeutics which allow us to repair this function come in the form of small molecules. In Chapter 5, we turn to the problem of predicting interactions between proteins and small molecules, also known as drug-target interaction (DTI). Our proposed solution uses PLMs in much the same way, widening the scope of which drug targets our model can be applied to. In addition, we tackle the challenging problem of decoy drugs using contrastive learning. This chapter is adapted from work originally published in:

- Singh, R.*, **Sledzieski, S.***, Bryson, B., Cowen, L., & Berger, B. (2023). Contrastive learning in protein language space predicts interactions between drugs and protein targets. Proceedings of the National Academy of Sciences, 120(24), e2220778120. [14]

Over course of the work presented in this thesis, PLMs have gone from a newly-introduced technology to a workhorse of computational molecular biology. However, their widespread adoption is still fundamentally limited by their computational expense. In Chapter 6, we take a first step towards the democratization of these models, by presenting methods for more efficient fine-tuning and adaptation of PLMs. This chapter is adapted from work originally published in:

- **Sledzieski, S.**, Kshirsagar, M., Baek, M., , Dodhia, R., Ferres, J. L. & Berger, B. (2024). Democratizing protein language models with parameter-efficient fine-tuning. Proceedings of the National Academy of Sciences, *In Press*. [15]

We conclude this thesis in Chapter 7, with a brief overview on the state of the field at the time of writing, including thoughts on potentially fruitful future directions. We also include several appendices (Chapters A, B, C, D), which contain algorithmic and experimental details held out of the main text for brevity and clarity.

1.4 Contributions of this thesis

We briefly summarize the primary contributions of this thesis towards the computational modeling of protein sequence & structure, protein interaction networks, cellular function, and therapeutic targeting of proteins.

- We introduce a novel machine learning architecture and training algorithm, leveraging for the first time protein language models to accurately predict PPIs in a wide variety of species, despite training only on known human interactions (Chapter 3).
- We apply this model towards advanced understanding of the metabolic process in the bovine rumen, through a newly constructed synthetic PPI network and computational analysis pipeline (Chapter 3).
- We develop a new training process for our machine learning model, incorporating a graph-theoretic analysis of the interaction network as data augmentation for training and improving the ability to perform transfer learning (Chapter 4).
- We contribute a new joint sequence-structure representation by including structure features as part of our framework, which enables our model to take advantage of new advances in computational prediction of protein structure when predicting PPI (Chapter 4).
- We introduce protein language models to the problem of drug-target interaction (DTI) prediction for the first time, developing a new machine learning model which generalizes better than previous state-of-the-art to out-of-distribution protein targets (Chapter 5).

- We develop the first method which uses contrastive learning to increase the specificity of our DTI model, yielding large gains in performance on highly difficult decoy discrimination tasks (Chapter 5).

- We apply our model for massive scale target-compound screens, evaluating over ten million candidates per day and identifying a putatively novel binder of *EPHB1* (Chapter 5).

- We introduce methods for parameter-efficient fine-tuning to protein language models for the first time, democratizing their use to resource-constrained labs and performing a deep evaluation of how to maximize performance under limited computation (Chapter 6).

Chapter 2

Background

Computational biology is by its very nature interdisciplinary, and as such the main chapters in this thesis assume some knowledge of both basic biological and computational concepts. In this chapter, we introduce several of these concepts, and provide the background necessary for the remainder of the work. This chapter is not meant to be a comprehensive primer into computational molecular biology, but rather a jumping off point for reading this thesis and beginning research in computational molecular biology. Keywords appear in **bold**.

2.1 The central dogma of biology

Although now known to be an oversimplification, the "central dogma" of biology, introduced by Watson and Crick in 1958 [16] remains a useful abstraction to understand information flow in the cell. Information in the cell is primarily stored in the form of large molecules called **deoxyribonucleic acid (DNA)**. DNA is composed of long strings of **nucleoside** bases (adenine [A], cytosine [C], thymine [T], and guanine [G]) joined by a sugar-phosphate backbone, jointly called a **nucleotide**. DNA is typically **double-stranded**, meaning two strings join with one another by complementary base-pairing rules (A-T, C-G) to form the classic helix.

These long strings form the genetic code. Substrings of the DNA, known as **genes** are

transcribed into **ribonucleic acid (RNA)**. RNA, like DNA, is composed of nucleotides, but is typically single-stranded and with thymine replaced with uracil [U]. While RNA has several cellular functions, **messenger RNA (mRNA)** act as copies of genes which can be transported throughout the cell without compromising the original DNA. Transcriptional regulation is one important method by which a cell controls its function.

RNA is **translated** into long chains of molecules called **amino acids** joined by **peptide bonds**; the resulting chain is called a **polypeptide** and long polypeptides are called **proteins**. Translation is performed by large molecules called **ribosomes**, which read off the mRNA in groups of 3 bases known as a **codon**. Each codon corresponds to one of 20 natural amino acids, each with a distinct **side-chain** and chemical properties (with some redundancy, there are $4^3 = 64$ distinct codons). Proteins, which are the primary focus of this thesis, perform a variety of cellular functions from transport of material within the cell, to signal transduction, to the very process of protein synthesis (transcription is performed by a protein called **RNA polymerase**).

2.2 Biochemistry and the relationship between sequence, structure, and function

Another classic, if simplified, notion of information flow within biology is the **sequence-structure-function** relationship [16]. This is the idea that the function of a protein is largely determined by its three-dimensional structure, and that this structure is in turn largely determined by the sequence of amino acids (**primary sequence**).

Though polypeptides are one-dimensional chains of amino acids, they undergo a folding process, first adopting common **secondary structures** such as **alpha helices** and **beta sheets**, then more complex **tertiary structures**. This folding process is driven by the formation of non-covalent chemical bonds between the side-chains, such as **hydrogen bonds**, **ionic bonds**, **hydrophobic interactions**, and in some cases covalent interactions such as

the formation of **disulfide bridges**. Proteins often adopt structure which are energetically favorable.

Often, multiple polypeptide chains will combine to form **quaternary structures** or **protein complexes**. Formation of these higher-order assemblies are likewise driven by non-covalent chemical interactions between amino acids, adopting energetically favorable conformations. Thus, the structure of these proteins determines their shape compatibility, and the strength with which they associate, known as their **affinity**. Because these chemical interactions are reversible, the affinity of an interaction refers to the ratio of bond formation to breaking, or the relative amount of time that a complex spends in a joined vs. un-joined state. In this thesis, when we refer to a **protein-protein interaction**, we are referring to this association between polypeptide chains, though we typically treat interactions as a binary measure. The function of a protein or complex, such as signaling, transport, or enzyme catalysis, is then determined by the interactions that it participates in, and the corresponding changes in protein shape and chemical properties.

We also refer to **protein-small molecule interactions** throughout this thesis. **Small molecules** can likewise participate in non-covalent interactions with a protein, and these interactions will change the shape and thus function of a protein (often by modulating its ability to bind with other proteins).

2.3 Algorithms for biological sequence analysis

The history of computational biology is rich with algorithms for analyzing DNA, RNA, and protein sequences. Throughout this thesis, we often refer to sequence similarity, which is typically computed by scoring an **alignment** between two sequences. This alignment accounts for evolutionary differences between sequences, including **point mutations**, **insertions**, and **deletions**. Needleman and Wunsch introduced an algorithm for **global** sequence alignment [17], and Smith and Waterman introduced a variation for **local** alignment [18] which requires

only sub-strings to match between the two sequences. The Basic Local Alignment Search Tool (**BLAST**) [19] was introduced for searching for searching a database for similar biological sequences, by first identifying a short matching "seed" sequence, and then extending the match.

Biological sequences are also commonly analyzed using **Markov models**. A **Markov chain** consists of a set of states and transitions between them, and have the property of being **memory-less**; that is, the probability of transitioning between two states is dependent only on the current state. For example, a simple Markov model of DNA sequences would be parameterized by an **initial distribution** over the four nucleotides, and a 4×4 **transition matrix** which indicates the probability of seeing any given (ordered) pair of nucleic acids.

A **hidden Markov model (HMM)** builds upon this idea, where the states no longer correspond to nucleotides (or in the case of protein sequences, amino acids), but are hidden and unknown *a priori*. Each hidden state has an associated set of **emission probabilities**, which are the likelihood that a given nucleotide will be generated by each state. Then, an HMM could generate sequences by sampling an initial state, and then at each time step emitting a nucleotide and transitioning to the next state. The initial distribution, transition probabilities, and emission probabilities can be learned from sequence data using the Baum-Welch algorithm, and the Viterbi algorithm [20] can be used to estimate which series of hidden states generated a specific sequence (note: these are not unique to biological sequences). **Profile HMMs** are the backbone underlying common sequence search tools such as HMMER [21] and hhsearch [22]. Recently, substantial advances in biological sequence modeling have been made using **large language models** (see Section 2.6).

2.4 Graphs and graph algorithms

The **graph** is a fundamental object in discrete mathematics and combinatorics. A graph is defined as a set of **nodes** or **vertices** representing discrete objects, and a set of **edges**

representing relationships between these objects. Edges can be **directed** or **un-directed** and optionally have a **weight** associated with them which typically corresponds to the strength of the relationship. Many networks, for example a social media network, can be represented as graphs, with each person as a node and "friendship" or "following" relationships as edges.

All nodes connected by an edge to a given node are known as the latter's **neighbors**. A **path** is an ordered list of edges starting at a **source** and ending at a **target**, and a **cycle** is a path which starts and ends at the same node. A graph is considered **connected** is there is a path between any two pairs of nodes, and large graphs will often have several **connected components**. A graph with no cycles is referred to as **acyclic**.

While graphs are defined as sets, they are often represented using **adjacency matrices (A)**, where each row or column represents a node, and the values of the matrix represent edge existence (0 or 1) or weight. This representation of a graph allows for easy computation of several interesting properties, such as node **degree**, or number of edges, which can be computed from row or column sums (note: adjacency matrices for directed graphs will often be asymmetric, with rows and columns representing outgoing and incoming edges). The **degree matrix (D)** is a diagonal matrix where each diagonal entry contains the degree of the corresponding node. The **Laplacian (L)** of a graph can be computed as $L = D - A$. The graph Laplacian can be used to compute various properties of the graph, including partitions and conductance, and is at the heart of community detection algorithms such as **spectral clustering**.

A **random walk** on a graph is a **path** starting at a **source** node where the next edge is selected probabilistically based on **transition probabilities**. These transition probabilities can likewise be represented as a matrix, and thus a random walk of length k can be computed by k successive matrix multiplications. Random walks typically are typically Markovian.

Common computational problems on graphs include community detection or node clustering, searches for shortest paths between two nodes, and **edge prediction**. It is the latter problem that we focus most heavily on in this thesis.

2.5 Machine learning and deep learning

The work in this thesis leans heavily on a subset of artificial intelligence known as **machine learning**. In contrast to traditional algorithms, machine learning algorithms are **data-driven**, meaning that their functions are derived from patterns within data (known as **training**), rather than being explicitly designed by the developer. Machine learning algorithms (**models**) are typically either supervised or unsupervised.

Supervised learning refers to tasks where the model is tasked to predict the **label** of a data point given some set of **features**; classical examples include identifying handwritten digits [23] or predicting the price of a home. In the former case, the features would be the values of the pixels in the images, while in the latter features may include things like the location of the home, the square footage, or the number of bedrooms.

Unsupervised learning refers to tasks where data are **unlabeled** and the goal is to organize or extract patterns from the data. A classical example of unsupervised learning is clustering, such as trying to identify groups of customers with similar shopping habits, where features might include demographic information or information on prior purchases.

A particularly successful subset of machine learning is **deep learning**, which makes use of a set of algorithms known as **neural networks**. Neural networks combine the input features using a combination of weighted sums and **non-linear functions**, where the model parameters or **weights** are learned from the data. These weights are incrementally updated through a process called **backpropagation** [24], which is based on the gradient of the **error**, or a measure of the model performance. The design of the network, including the number of parameters and the methods of feature combinations, is typically referred to as the **architecture**.

While this thesis is primarily focused on supervised learning, we also rely very heavily on a set of unsupervised techniques known as **representation learning**. In representation learning, the objective is to learn some lower-dimensional vector representation of a high-

dimensional data point (commonly called an **embedding**) such that some notion of similarity is preserved. For example, vector embeddings of words would be similar if the words have related meanings or grammatical functions, and embeddings of images would be similar if the images depict similar subjects. These embeddings can be learned in an unsupervised manner, in a process known as **pre-training**, and then used as a set of features for supervised learning tasks.

2.6 Large language models

Large language models (LLMs) are a subset of neural networks which are designed to model sequences, generally text split up into smaller chunks called **tokens**. Tokens are typically a word or sub-sequence of a word (e.g. a suffix like "-ing"). There are several LLM architectures, including the long short-term memory network (LSTM) [25] and bidirectional LSTM (bi-LSTM) [26], but most modern LLMs use variants of an architecture called the **transformer** [27], which uses a mechanism known as **attention** to aggregate information between tokens. LLMs are typically pre-trained to learn representations of text, either by trying to predict the text token given a previous set (known as **auto-regressive** or **causal** language models), or by trying to predict which tokens are held out from the middle of text (known as **masked** language modeling). Neural language models are based on the **distributional hypothesis** [28], which proposes that words in similar contexts have similar meanings. In this work, we use **protein language models (PLMs)** which are trained on protein sequences where each token is an amino acid. Protein sequences largely conform to the distributional hypothesis, where the function of an amino acid is determined by the sequence context and the chemical bonds that it is able to form.

Chapter 3

Protein-Protein Interactions I: Sequence

3.1 Chapter Overview

Machine learning models for predicting protein-protein interactions (PPI) often fail to generalize well between species, and are limited by the dearth of known interactions to train on in many species. In addition, many proteomic analyses are limited to only the known sequence, without substantial information about the structure of the protein. In this chapter, we introduce a new deep learning called D-SCRIPT, for predicting a physical interaction between two proteins given just their sequences. It generalizes well to new species and is robust to limitations in training data size. Its design reflects the intuition that for two proteins to physically interact, a subset of amino acids from each protein should be in contact with the other. The intermediate stages of D-SCRIPT directly implement this intuition, with the penultimate stage in D-SCRIPT being a rough estimate of the inter-protein contact map of the protein dimer. This structurally-motivated design enhances the interpretability of the results and, since structure is more conserved evolutionarily than sequence, improves generalizability across species.

3.2 Introduction

The systematic mapping of physical protein-protein interactions (PPIs) in the cell has proven extremely valuable in deepening our understanding of protein function and biology. In species such as yeast and humans where a large network of experimentally determined PPIs exists [29]–[33], this PPI network information has proven valuable for downstream inference tasks in understanding functional genomics and biological pathway analysis [34]–[38]. However, in most species— especially, non-model organisms— the coverage of experimental PPI data remains very low (Figure 3.1a). There, computational prediction of PPIs can help mitigate the lack of experimental data and facilitate biological discovery. While substantial progress has been made in PPI prediction overall, the important case of *de novo* prediction for less-studied proteins and non-model organisms continues to be a challenge. The lack of functional genomic data in such situations makes it difficult to apply methods based on bootstrapping from the connectivity patterns of known PPIs [39]–[42] or those that infer PPIs from other protein-protein association modalities like co-expression and co-localization [36], [43]–[45]. Recently, deep learning-based methods have offered the prospect of predicting PPIs just from sequence data. Unfortunately, existing models [46], [47] have shown limited generalizability: they work quite well when applied to the species they were trained on, but their performance declines in a cross-species context.

Here, we introduce D-SCRIPT (Deep Sequence Contact Residue Interaction Prediction Transfer), a structure-aware deep learning approach to PPI prediction. It has a novel, geometrically interpretable neural-network architecture that we show is able to make meaningful PPI predictions in the cross-species setting. Our key conceptual advance is implementing an interpretable, structure-based model despite only having sequence-based inputs: a well-matched combination of input featurization and neural-network architecture allow for D-SCRIPT to be trained solely from sequence data, supervised only with a binary interaction label, and yet produce an intermediate representation that substantially captures the structural

mechanism of interaction between the protein pair. Leveraging recent advances in deep learning-based language modeling of proteins, we first apply Bepler and Berger's [48] deep learning-based language model of single protein sequences to construct our features. Using this pre-trained model results in informative protein embeddings (i.e., representations in a high-dimensional space) that are implicitly endowed with structural information about each of the proteins. D-SCRIPT's generalizability and interpretability then comes from its ability to learn informative geometric representations of the proteins, individually and jointly. In particular, it learns how to transform the two protein embeddings into a 2-D contact map, encoding the intuition that a physical interaction between two proteins requires that a subset of the residues in each protein be in contact with the other protein (Figure 3.1b,c).

Evaluating D-SCRIPT in the cross-species prediction setting, where a method trained on human PPIs is used to predict PPIs in several less-studied model organisms, we show that it substantially improves upon existing methods, including the state-of-the-art deep learning method PIPR [47] in a stringent cross-validation experiment. In addition to comparing the accuracy of PPI predictions in cross-validation, we demonstrate the interpretability and downstream utility of D-SCRIPT results in several ways. First, we demonstrate that on a genome-wide scale, *de novo* PPIs predicted by D-SCRIPT produce a network whose modular structure produces clusters of proteins with greater functional coherence than those produced from PIPR predictions. Next, on assessing the physical plausibility of the intermediate contact-map representation, we find that the map partially discovers the structural mechanism of an interaction despite the model having been trained only on sequence data. Specifically, we evaluate our predictions on Hwang et al.'s [49] benchmark database of 3-D structures of docked protein-pairs and observe that our model's predicted contact map is substantially similar to the ground-truth inter-protein contact map in cases where our model predicts an interaction.

To demonstrate the utility of D-SCRIPT as a tool to study novel systems in less-studied organisms, we investigate the rumen in *Bos taurus* (cow). We apply D-SCRIPT to predict

new PPIs across a large subset of bovine proteins and decompose the network of D-SCRIPT predicted PPIs into functional gene modules. Starting from a seed set of genes found to be over-expressed in the rumen, we identify five functional gene modules involved in cellular metabolism and growth, immune response, and transcriptional regulation, suggesting links between metabolism and transcriptional regulation through *MRPL4* and *H15* domain containing proteins.

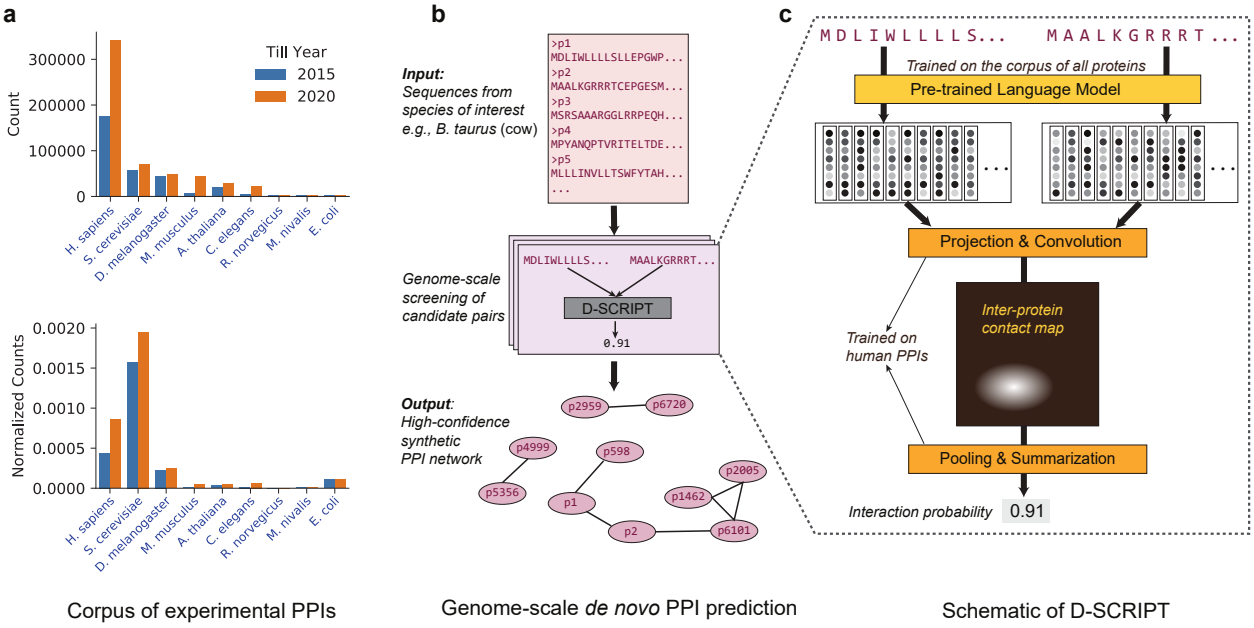


Figure 3.1: **D-SCRIPT Motivation and Workflow.** We demonstrate how D-SCRIPT can be used genome-wide to predict a complete PPI network in the fly. **(a)** Experimentally derived PPI data is scarce in species outside of human and yeast, even when normalized for size of the genome. (sourced from BioGRID, see Section A.1 for details). **(b)** A D-SCRIPT model, after being trained on a large corpus of human PPI data, can be broadly applied to a species of interest, even if little PPI data is available in that species. For each pair of proteins in the target species, D-SCRIPT converts the pair of protein sequences into a score representing probability of interaction. Because D-SCRIPT scales to large numbers of protein pairs and maintains performance across species, it can be used to score all protein pairs genome-wide to predict a synthetic PPI network in the species. **(c)** Blowup detail of the D-SCRIPT architecture from the box in (b) (and see Figure 3.2 for more detail). D-SCRIPT generalizes due to its structurally motivated design. The pre-trained language model generates structural features for a single protein, while the projection and convolution model the interaction between every pair of residues in the candidate pair. The novel regularization ensures the prediction of an inter-protein contact map which is sparse with a few high probability contacts during prediction of interaction.

3.3 Methods

3.3.1 PPI data set

To evaluate the performance of D-SCRIPT in predicting protein-protein interactions, we use data from the STRING [44] database (version 11). STRING contains protein pairs corresponding to a variety of primary sources and interaction modalities (e.g., binding vs co-expression). In order to select only high-confidence physical protein interactions, we limited our positive examples to binding interactions associated with a positive experimental-evidence score. From this set, we removed PPIs involving very short proteins (shorter than 50 amino acids) and, due to GPU memory constraints, also excluded proteins longer than 800 amino acids. Next, we removed PPIs with high sequence redundancy to other PPIs, following the precedent of previous approaches [47], [50]. Specifically, we clustered proteins at the 40% similarity threshold using CD-HIT [51], [52], and a PPI (A-B) was considered sequence redundant (and excluded) if we had already selected another PPI (C-D) such that the protein pairs (A, C) and (B, D) each shared a CD-HIT cluster. Removing sequence redundant PPIs from the data set prevents the model from memorizing interactions based on sequence similarity alone. To generate negative examples of PPI, we followed [50] and randomly paired proteins from the non-redundant set, choosing a 10:1 negative-to-positive ratio to reflect the intuition that true positive PPIs are likely rare. Our human PPI data set contained 47,932 positive and 479,320 negative protein interactions, of which we set apart 80% (38,345) for training and 20% (9,587) for validation. For each of 5 model organisms (Table 3.1) we selected 5,000 positive interactions and 50,000 negative interactions using this procedure, with the exception of *E.coli* (2,000/20,000) where the available set of positive examples in STRING was limited.

3.3.2 Overview of D-SCRIPT

Our deep learning model for predicting PPIs directly from protein sequences, similar to previous deep learning methods DPPI [46] and PIPR [47], is composed of two stages (Figure 3.1c). The first stage generates a rich feature representation for each protein separately and the next stage estimates an interaction probability based on these features, with the model being trained end-to-end across both stages. In both DPPI [46] and PIPR [47], much of the model complexity lies in the feature generation, which is learned *ab initio* from the training data.

D-SCRIPT differs from these approaches in the design and relative complexity of the two stages. First, we apply a pre-trained model to generate rich, structurally-informative feature representations of the proteins (Figure 3.2). The pre-trained model was developed by Bepler and Berger [48], [53], who built upon advances in deep learning-based modeling of natural languages to design a language model for protein sequences: an n-amino acid protein sequence is mapped to an $n \times 6165$ representation, with the various dimensions capturing local and global aspects of the protein structure. We then learn a lower-dimensional projection ($n \times 100$) of this embedding as a compact representation for downstream interaction and structural prediction tasks. The second stage of D-SCRIPT encodes a structure-based model of protein interaction: in the contact module, the low-dimensional embeddings are used to compute an inter-protein contact map that corresponds to the locations of residue contacts between protein structures, and in the interaction module, this contact map is summarized into a single score (i.e., the probability of interaction). In each layer, the mathematical operations performed are rooted in structural intuitions. For example, to formalize the intuition that true-positive contact maps should be sparse but have isolated regions of strong contacts, we introduce a customized max pooling operation and a novel regularization loss function. A more detailed description of our model architecture and training process is provided in Section 3.3.

3.3.3 Model architecture

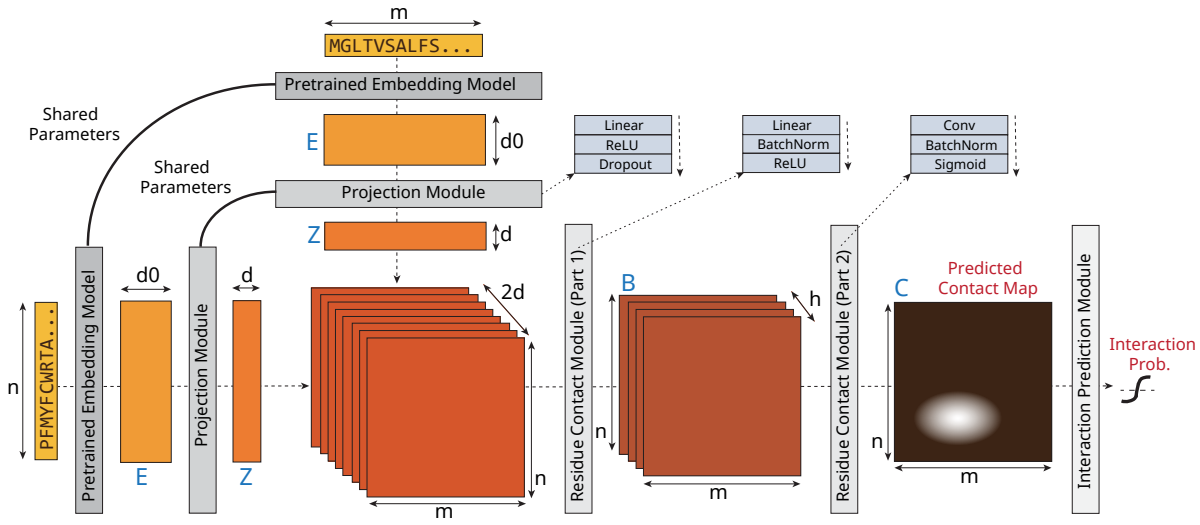


Figure 3.2: **D-SCRIPT Architecture** Left to right: The Pre-trained Embedding Model generates features for each individual protein. The Projection Model reduces them to d dimensions. Each low-dimensional single-protein embedding implicitly includes, among other features, an encoding that broadly captures the protein's residue-contact map (Figure 3.4). The novel portion of the architecture combines these low-dimensional embeddings to compute a sparse *inter-protein* contact map through a two-step process which first computes a representation for each pair of residues, then incorporates local information using a convolutional filter. Finally, the interaction prediction module uses a customized max-pooling operation to predict the probability of interaction between the input proteins.

Input & Structure-aware Embedding The input to D-SCRIPT is a pair of protein sequences S_1, S_2 with lengths (n, m) and it outputs an interaction probability $\hat{p} \in [0, 1]$ and a predicted-contact matrix $\hat{C} \in [0, 1]^{n \times m}$.

We first generate embeddings $E_1 \in \mathcal{R}^{n \times d_0}, E_2 \in \mathcal{R}^{m \times d_0}$ by embedding S_1 and S_2 with a pre-trained model from Bepler and Berger. Their model is a Bi-LSTM (bidirectional long short-term memory) neural network trained on three independent pieces of information: 1) the protein's SCOP classification, indicating its general structure, 2) self-contact map of a protein's 3-D structure, and 3) sequence alignment of similar proteins. These embeddings capture both local and global structural features of the protein sequences: the d_0-dimensional encoding of each amino acid contains information not just about the amino acid and its

immediate context, but also the global structure of the protein. This is a key distinction from other approaches (e.g. Chen et al.'s in PIPR), where each amino acid's embedding represents just its biochemical properties or a short-range context (e.g., 5-7 residues) around it. We note that alternative embeddings (e.g. Rives et al. [54] or Luo et al. [55]) can potentially be substituted here.

Projection Module In the projection module, we reduce E_1 and E_2 to d-dimensional representations using a fully-connected linear layer (multi-layer perceptron) with d_0 input and d output nodes. Specifically, given an input embedding $E \in \mathcal{R}^{n \times d_0}$, we compute the embedding projection $Z \in \mathcal{R}^{n \times d}_{\geq 0}$ as

$$Z_i = \text{Drop}(\text{ReLU}(E_i W_1 + b_1)) \; \forall i \in 1 \ldots n \tag{3.1}$$

with $W_1 \in \mathcal{R}^{d_0 \times d}, b_1 \in \mathcal{R}^d$ as learned weights and biases. The rectified linear unit (ReLU) is a non-linear operation which applies the transformation $\text{ReLU}(x) = \max(0, x)$. The dropout layer (Drop) randomly sets 50% of the weights to zero, helping prevent over-fitting in W_1.

Residue Contact Module The residue contact model takes the d-dimensional embeddings Z_1, Z_2 and models the interaction between the residues of each protein. First, for each pair of residue embeddings $Z_{1_i}, Z_{2_j} \in \mathcal{R}^d, i \in 1, \ldots, n, j \in 1, \ldots, m$, we compute a broadcast matrix with hidden dimension h, $B_{z_0, z_1} \in \mathcal{R}^{n \times m \times h}_{\geq 0}$

$$\text{diff}_{i,j} = |Z_{0_i} \ominus Z_{1_j}| \tag{3.2}$$

$$\text{mul}_{i,j} = Z_{0_i} \odot Z_{1_j} \tag{3.3}$$

$$B_{i,j} = \text{ReLU}(\text{Batch}([\text{diff}_{i,j}, \text{mul}_{i,j}] W_2 + b_2)) \tag{3.4}$$

where $\ominus$ indicates the element-wise difference and $\odot$ indicates the Hadamard product.

This featurization is symmetric and has been previously used in NLP and protein sequence modeling tasks [48], [53], [56].

$W_2 \in \mathcal{R}^{2d \times h}$, $b_2 \in \mathcal{R}^h$ are the learned weights and biases. The batch normalization operation normalizes the mean and variance of the input features, thus stabilizing the learning process. Each element $B_{i,j}$ captures the direct interaction between residues S_{1_i} and S_{2_j}. The broadcast matrix B is used to compute the contact prediction matrix $\hat{C} \in [0,1]^{n \times m}$, where

$$\hat{C}_{i,j} = \sigma(\text{Batch}(\text{Conv}(B_{(i-w:i+w),(j-w:j+w)}))) \tag{3.5}$$

The two-dimensional convolution (Conv) operation with width $2w+1$ and h channels uses the h-dimensional representation of all residues within w of $B_{i,j}$ to compute $\hat{C}_{i,j}$, and thus detects local patterns in two-dimensional residue contact space. The broadcast matrix is zero-padded to allow for the convolution operation to be performed at all indices. We again apply a batch normalization to stabilize learning. We apply the sigmoid operation σ, which restricts the output values of $\hat{C}$ to be in the range [0,1], and thus they can be interpreted as the predicted probability that two residues are in contact.

Interaction Prediction Module The interaction prediction module computes a single probability of interaction $\hat{p}$ from the $n \times m$ contact map $\hat{C}$. To do so, we perform two pooling operations. The first is a standard max-pool: an l-dimensional max-pool divides $\hat{C}$ into $\lceil \frac{n}{l} \rceil \times \lceil \frac{m}{l} \rceil$ non-overlapping regions and takes the maximum value of each region, with zero-padding applied where necessary. This max-pooled matrix P represents the probability of interaction in local regions of the proteins and maintains only the highest-probability residue contacts in each region for global prediction. The second pooling operation is a global pooling operation, calculated as

$$Q_{i,j} = \text{ReLU}(P_{i,j} - \mu - (\gamma * \sigma^2)) \tag{3.6}$$

$$\hat{p}_{raw} = \frac{\sum_{i,j} Q_{i,j}}{\sum_{i,j}(\text{sign}(Q_{i,j})) + 1} \tag{3.7}$$

where μ, σ^2 are the mean and variance of the P_{ij} values and γ is a learned parameter. The matrix Q sparsifies P, maintaining only those contacts which are $\gamma\sigma^2$ greater than the mean, and setting all others to zero. We then predict that the proteins will interact with the average probability of interaction among these high-probability contacts. Together with the regularization that the contact matrix be sparse, this global pooling operation captures the intuition that a pair of interacting proteins will be characterized by a relatively small number of high-probability interacting residues or regions.

The final step of interaction prediction is designed to enhance the bimodality of the output distribution, so that the choice of a cutoff becomes less important in distinguishing positive and negative predictions. We apply the logistic activation function to compute the output probability $\hat{p} = \sigma_{(x_0,\eta)}(\hat{p}_{raw})$ where

$$\sigma_{(x_0,\eta)}(x) = \frac{1}{1 + e^{-\eta(x-x_0)}} \tag{3.8}$$

This activation function, with $x_0 = 0.5$, takes our raw probability predictions and makes them more "extreme", depressing values below x_0 towards 0 and inflating values above x_0 towards 1, with η controlling the rate at which this occurs. We return $\hat{p}$ and $\hat{C}$ as the model prediction, from which we calculate the loss and optimize the gradient.

3.3.4 Training

Training Objective Given the true labels, the predicted probabilities $\hat{p}$, and the contact maps $\hat{C}$, we compute the loss as $\lambda L^{BCE} + (1-\lambda)L^{MAG}$; here λ is a hyper-parameter that balances between L^{BCE}, the binary cross-entropy (BCE) loss, and L^{MAG}, the contact-map magnitude loss (MAG). While the BCE loss is standard in a classification context, we introduce L^{MAG} as a novel regularization that enables us to learn realistically sparse contact

maps. L^{MAG} for a single training example is calculated as the arithmetic mean value of the contact map $\hat{C}$. Jointly minimizing the total magnitude of contact maps with the BCE captures the intuition that interacting proteins are characterized by just a few high probability inter-protein contacts, while most residues will not be in contact.

Implementation Details We implemented D-SCRIPT in PyTorch 1.2.0 and trained with a NVIDIA Tesla V100 with 32GB of memory. Embeddings from the pre-trained Bepler and Berger model were produced by concatenating the final values of the output and all hidden layers, so that $d_0 = 6165$. We used a projection dimension of $d = 100$, a hidden dimension of $h = 50$, a convolutional filter with width $2w + 1 = 7$, and a local max-pooling width of $l = 9$. We used $x_0 = 0.5, \eta = 20$ for the custom logistic activation, and $\lambda = 0.35$ for calculating the training loss. Weights were initialized using PyTorch defaults. We used a batch size of 25, the Adam optimizer with a learning rate of 0.001, and trained all models for 10 epochs.

3.3.5 Assessing functional module coherence

Detection of protein functional modules was performed by spectral clustering, with pairwise distances between proteins assessed using Cao et al.'s [57], [58] diffusion state distance (DSD) metric. We generated 500 clusters, removed clusters with fewer than 3 nodes, and recursively split clusters with greater than 100 nodes. This module detection approach performed well in a recent DREAM challenge on functional module detection [38]. Proteins were annotated with functions using Gene Ontology (GO) terms from FlyBase [59], filtering out electronically-inferred and homology-based annotations. All GO terms were mapped to a limited set of GO Slim terms using the *D. melanogaster* species-specific list [60]. For each cluster, we computed the within-cluster functional similarity, calculated as the mean Jaccard similarity of the sets of GO Slim annotations for all pairs of proteins within the cluster. We also used a different, graph-theoretic measure of protein similarity based on GO terms [61]; it produced similar results. The distribution of within-cluster similarity scores were compared using a one-tailed

t-test, with the null hypothesis that the 374 modules in the PIPR network and 384 modules in the D-SCRIPT network have the same average within-cluster similarity.

3.3.6 Predicting PPIs in *Bos taurus*

We selected 24,195 bovine proteins by selecting the longest isoform of each gene and, using HMMER [62] and GODomainMiner [63], limited ourselves to proteins that had at least one PFAM [64], [65] domain with some associated GO annotation. In general, this filtering step is unnecessary but here it allowed us to focus our computational resources on proteins likely to be interest. Of the set of 292.7 million possible pairwise combinations, we generated a list of fifty million candidate interactions by randomly sampling protein pairs; we included a special check to ensure that the 12 pre-identified rumen related genes were fully covered. We predicted the probability of interaction using the human-trained D-SCRIPT model, and selected edges with a predicted value greater than or equal to 0.5 as positive edges. This resulted in a predicted network of 476,399 positive interactions between 17,811 proteins with an average node degree of 53.5. We created functional protein modules by performing spectral clustering on the proteins, using the diffusion state distance (DSD) [57] metric. For each module, we applied gene set enrichment analysis using the g:GOSt tool on the g:Profiler web server [66] to identify functions over-represented among the proteins in each cluster.

We analysed bovine gene expression using bulk RNA-seq data from 93 tissue-specific bovine samples [67] (GEO Accession GSE160028) and normalized each sample to counts per million (CPM). For each cluster, we identified genes which are expressed more highly in rumen tissue than on average across all tissues, and compute and report the log fold increase in expression in the rumen. The gene expression data was also used to verify the quality of our predicted network and modules. We computed the Spearman rank correlation between the CPM normalized expression profiles for each pair of genes, then evaluated the average correlation between protein pairs with no interaction predicted, pairs with a predicted positive edge, pairs which are predicted to interact and share a cluster, and pairs of proteins which

appear in the same module, regardless of whether or not an edge appears. We perform a one-sided Welch's t-test to compare sample means without assuming equal variance.

3.3.7 Identifying the catalytic signature motif of bovine protein tyrosine phosphates

Our analysis of functional modules in the bovine rumen found a single cluster containing 34 proteins homologous to human protein tyrosine phosphatases (PTPs). PTPs are known to comprise several families with similar sequence but diverse binding specificity, determined in part by a short catalytic signature motif [68], [69]. In Figure 3.8, we show Cluster A from Figure 3.7a, recolored based on the PTP sub-type. All but one of the neighbors of PRD-SPRRII in Cluster A are PTP proteins (the exception being MARK2, a serine/threonine-protein kinase). We find that D-SCRIPT does not limit its predictions to one family, instead predicting interactions in all sub-types.

To further investigate how D-SCRIPT determines binding specificity, we undertook an in silico mutagenesis experiment. The canonical catalytic signature motif for the PTP family is HCX_5R [70] or HCXXGXXR [69], a motif which we identified in 28 PTP proteins from Cluster A. We also included ENSBTAP00000067545 (CDC25C), which has the motif HCX_5A in our analysis. For each protein, we used D-SCRIPT to predict the probability of interaction with ENSBTAP00000070493 (PRD-SPRRII). Then for 50 replicates, we randomly perturbed the catalytic motif in that protein, either by randomly selecting amino acids for all 8 positions of the motif, or only the 5 flexible positions (X). We find that for 24 of the 29 proteins, perturbing only the flexible positions does not reduce the D-SCRIPT predicted probability, while perturbing the entire motif drastically decreases the predicted probability. For 2 of the remaining 5, D-SCRIPT already did not predict an interaction with the original protein, for another one perturbing even the flexible positions decreased the probability of interaction substantially, and for the final 2 even perturbing the entire motif did not substantially decrease the predicted probability of interaction. Figure 3.9a shows

the original prediction (red), the distribution of 50 replicates where only flexible sites were mutated (blue), and the distribution of 50 replicates where the entire catalytic motif was mutated (orange) for each PTP protein. Finally, we sought to identify which residues were most important in determining the D-SCRIPT model's prediction. To do so, we selected 5 CDC14 proteins (ENSBTAP00000058782, ENSBTAP00000073534, ENSBTAP00000054725, ENSBTAP00000069880, ENSBTAP00000070948) and aligned them using MUSCLE [71]. Then, for each position in the alignment, for all sequences which didn't have a gap in that position, we randomly perturbed the amino acid at that position 50 times and used D-SCRIPT to predict interaction between the perturbed sequence and PRD-SPRRII. For each sequence location, we computed the difference between the original predicted probability of interaction and the average probability of interaction predicted for the samples perturbed at that location, averaged across all 5 sequences (Figure 3.9b). We found the largest decreases around the catalytic signature motif, indicating that D-SCRIPT is in fact basing its predictions on the residues involved in binding specificity. Further, when we zoom in to the 8-residue motif region, it is clear that D-SCRIPT is identifying the conserved "C" and "R" in the second and eighth position, and the flexible "A" in the fourth position of the motif as the most important residues for determining interaction. Figure S5c shows the WebLogo for this region in the PTP domain, where the y-axis is the change in predicted probability when each position is perturbed [72], [73].

3.3.8 Predicting PPIs in SARS-CoV-2

We performed a preliminary study to predict viral-host interactions between SARS-CoV-2 and human proteins wherein we compared the sets of over-represented GO terms for human interactors of SARS-CoV-2 proteins, as predicted by D-SCRIPT or PIPR, with those over-represented in the experimentally-determined human interactors [74]. Figure 3.10 shows the relative similarity of computationally predicted annotations to the experimentally-determined annotations for each SARS-CoV-2 protein. Overall, we found that sets of enriched terms

computed using the D-SCRIPT network overlap slightly more with the true network than those computed using the PIPR network (p=0.059). Among the putative accessory factors (ORF* and Protein 14), D-SCRIPT performs significantly better (mean Jaccard similarity 0.029 vs. 0.118, p=0.022, paired one-tailed t-test). Visually, PIPR seems to be somewhat better at predicting interaction partners for the non-structural proteins (NSP*), although D-SCRIPT still has a slightly larger mean similarity (0.183 vs. 0.222, p=0.221). While D-SCRIPT performs better on the intensively studied spike (S) protein, PIPR shows a higher overlap for the nucleocapsid (N). Neither method predicts enriched terms for the other structural proteins encoding the envelope (E) and membrane (M) (0.149 vs. 0.121, p=0.672 across the four proteins).

Candidate pairs were generated using the viral sequences from Gordon et al. [74] and 19,777 human sequences from the STRING database, and predicted edges using D-SCRIPT and PIPR. We predicted 3,273 edges using D-SCRIPT and 2,922 edges using PIPR. 332 putative true viral-host interactions were taken from Gordon et al. Human sequences were mapped to UniProt sequences identifiers from Gordon et al. with sequence similarity $\geq 95\%$ using BLAST [19], and UniProt identifiers were used to identify a set of Gene Ontology terms for the human interactors of each viral protein. Following Gordon et al., we identified over-represented GO terms using the clusterProfiler R package (version 3.14.3) [75] with a 1% false discovery rate (FDR). Over-represented GO terms were mapped to a common set of terms taken from the ChEMBL Drug Target GO Slim Subset [76]. For each viral protein, we computed the Jaccard similarity between the set of GO Slim terms enriched in the putative true network and each of the computationally predicted methods. We computed a paired one-tailed t-test to statistically compare the relative similarities of D-SCRIPT and PIPR. Virus-host edges predicted using D-SCRIPT or PIPR are available for use by the community.

3.3.9 Logistic regression for prediction of protein structure

We selected 300 proteins with structural coordinates from the Protein Data Bank (PDB), and randomly split them into a training set of 100 and test set of 200 proteins. We assessed intra-protein contacts at 8 Å in the PDB structure and converted each protein's contact map to a binary classification data set: a protein sequence of length n corresponded to $\frac{n(n-1)}{2}$ observations, with the observation ij corresponding to a putative contact between residues i and j. Features were generated using the human pre-trained D-SCRIPT model, where we evaluate the model up through the first stage (projection module). This generates $d = 100$ dimensional vector representations Z_i and Z_j output by the projection module for each pair of amino acids in the protein which capture both local and global structural features. The regression was L_2-regularized and class balanced, with its input for observation ij being the concatenation of the 100-dimensional embeddings as well as their combinations diff_{ij}, mul_{ij} as defined in Equations 3.2 & 3.3.

3.3.10 Predicting interactions by nearest neighbor search

To compare our sequence embedding method with other potential ones, we evaluated various protein sequence embeddings under the following framework: the Euclidean distance in an embedding space was used to define a distance measure between proteins. Given a true positive PPI (A, B), we applied this distance measure to identify the k-nearest neighbors of A and B each, and computed how many of the k^2 possible combinations of these neighbors corresponded to a positive PPI. For D-SCRIPT, we used the $\mathcal{R}_{\geq 0}^{n \times d}$ output of the projection module and averaged the features across the length of the protein to obtain a d-dimensional embedding. For AAClass, Vec5, and the random embedding, we directly compute the Euclidean distance between those vectors. To identify k nearest neighbors using BLAST, we create a database of all other proteins in the species, perform a search using `blastp` with the default values, and return the top k hits by e-value.

3.3.11 Predicting inter-protein contact maps

The output of the residue Contact Module of D-SCRIPT is an inter-protein contact map $\hat{C}$ where $\hat{C}_{ij} \in [0, 1]$ can be interpreted as the probability of residue i from protein S_1 being in contact with residue j of protein S_2. We interpreted ground truth and predicted contact maps as probability distributions over the $n \times m$ matrix and measured the 2-D Earth mover's distance between these distributions, computed by solving an optimal transport problem under the Euclidean metric [77]. For each candidate PPI, we established random baselines by shuffling $\hat{C}$ 500 times and recomputing the Earth mover's distance between the random shuffle and the true contacts. We assign a p-value to each candidate PPI based on this permutation test and the probability of seeing an Earth mover's distance at least as small as the observed distance. We compute an overall p-value for positive-predicted pairs by computing a one-tailed t-test with the null hypothesis that the average candidate PPI p-value is 0.5, i.e. that the predictions are as accurate as a random shuffling.

3.3.12 Deconstructing model performance by protein frequency in training data

To further investigate the relative performance of each model on out-of-species classification, we evaluated each model on subsets of proteins ranked by their frequency in the training set. For a set of quantiles $q \in 0.5, 0.6, 0.7, 0.8, 0.9, 1.0$, we evaluated out-of-sample D-SCRIPT and PIPR predictions on the human PPI sub-network consisting only of proteins of quantile q or lower; here, lower q corresponds to a lower frequency of occurrence. In absolute terms, as Table 3.3 indicates, both D-SCRIPT and PIPR become more accurate at higher q. However, D-SCRIPT has a relative advantage at lower q (i.e., infrequently occuring proteins) while PIPR performs better at higher q. In other words, PIPR's better within-species performance can be traced to it being more accurate on proteins that occur frequently. This also suggests an explanation for PIPR's lower cross-species generalizability than D-SCRIPT: when making

predictions on an entirely new set of proteins in a different species, knowing the relative frequencies of proteins in the training data might not be particularly useful.

The difference between D-SCRIPT and PIPR might stem from their respective architectures. The protein representation learned by D-SCRIPT is constrained to be a linear projection of the Bepler and Berger pre-trained embedding, albeit with ReLU and dropout layers. This regularizes how much frequency information can be incorporated into the model; we note that the Bepler and Berger model was trained with data on individual proteins and would not reflect PPI frequency information. In contrast, PIPR's design allows for a lot more leeway in training each protein's representation. This flexibility may allow PIPR to better incorporate the occurrence frequencies into its representation, helping its within-species performance but potentially hurting its cross-species generalizability.

3.3.13 Quantification and statistical analysis

Statistical tests were conducted using version 1.3.1 of the SciPy Python package.

3.4 Results

3.4.1 D-SCRIPT generalizes well across species

We first sought to see how D-SCRIPT performed on the task of cross-species interaction prediction. We trained a model on human PPIs and evaluated it using PPI data sets from five other model organisms (Section 3.3). We compared D-SCRIPT with PIPR, shown by Chen et al. [47] to be currently the best performing sequence-based PPI prediction method, training both models on the same set of human PPIs; we compare their model complexity in Section A.2. In Table 3.1, we report the precision, recall, area under precision-recall curve (AUPR) and area under ROC curve (AUROC) of each method in each of five species. For highly unbalanced data, as is the case here, we note that AUPR is generally considered a

better metric than AUROC. D-SCRIPT outperforms PIPR in a cross-species setting and maintains a high AUPR across all species, even those which are extremely evolutionary distant from human. In fact, its AUPR in these species remains comparable to that seen in human cross-validation. Additionally, we compared with a hybrid method (D-HYBRID) where PIPR was used to adjust D-SCRIPT's prediction: when PIPR is highly confident that an interaction *does* occur ($\hat{p} > 0.9$), the predicted probability of D-SCRIPT is increased by 50%. The hybrid method outperforms both D-SCRIPT and PIPR alone, but improves D-SCRIPT only modestly in cross-species analysis. We also compare to a recently released method by Richoux et al. [78] in Section 3.4.2.

While our focus is on enhancing cross-species PPI prediction quality, we also sought to investigate how D-SCRIPT would perform at predicting within-species interactions in Human.

We performed 5-fold cross validation and report here the average across all folds. Additionally, we evaluated a hybrid method (D-HYBRID) where we used D-SCRIPT to adjust PIPR's prediction: when D-SCRIPT is highly confident that an interaction does *not* happen ($\hat{p} < 0.01$), we reduce the predicted probability from PIPR by half. Table 3.1 shows that while PIPR outperforms D-SCRIPT on human PPIs in cross-validation, a combination of the methods outperforms either one alone. Notably, D-HYBRID achieves substantially higher precision, though at the expense of recall. This may be a desirable trade-off in certain contexts, e.g., when generating PPI candidates for experimental validation. We note that while the incremental performance of the hybrid models is modest, the observation that D-SCRIPT performs better on out-of-sample species while PIPR performs better on in-sample species makes possible a simple combination of the two that does not substantially increase computation time, yet results in more accurate predictions both across and within species.

Table 3.1: **Evaluation of models trained on human PPIs.** *H. sapiens* results are average performance over 5-fold cross validation. All other species were evaluated using a model trained on human data. D-SCRIPT strongly outperforms PIPR cross-species, despite PIPR performing better in human cross-validation. D-HYBRID refers to D-SCRIPT confidence adjusted using PIPR predictions, and P-HYBRID refers to PIPR confidence adjusted using D-SCRIPT predictions. In both cases, a hybrid model is able to achieve better performance than either model alone.

Species	Model	AUPR	Precision	Recall	AUROC
M. musculus	PIPR	0.526	0.734	0.331	**0.839**
	D-SCRIPT	0.580	0.818	0.346	0.833
	D-HYBRID	**0.609**	**0.820**	**0.355**	0.838
D. melanogaster	PIPR	0.278	0.521	0.121	0.728
	D-SCRIPT	0.552	**0.798**	0.359	**0.824**
	D-HYBRID	**0.562**	**0.798**	**0.361**	**0.824**
C. elegans	PIPR	0.346	0.673	0.142	0.757
	D-SCRIPT	0.548	0.840	0.306	0.813
	D-HYBRID	**0.559**	**0.841**	**0.308**	**0.814**
S. cerevisiae	PIPR	0.230	0.398	0.085	0.718
	D-SCRIPT	0.405	0.706	0.223	**0.789**
	D-HYBRID	**0.417**	**0.708**	**0.225**	**0.789**
E. coli	PIPR	0.308	0.629	0.131	0.675
	D-SCRIPT	0.571	0.791	**0.520**	**0.863**
	D-HYBRID	**0.588**	**0.793**	0.394	**0.863**
H. sapiens (5-fold cross validation)	PIPR	0.835	0.838	**0.701**	0.960
	D-SCRIPT	0.516	0.728	0.278	0.833
	P-HYBRID	**0.844**	**0.949**	0.400	**0.962**

3.4.2 Comparison with another sequence-based method

We also compared our method to a model described by Richoux et al. [78]. In it, they trained a multi-layer fully connected network to predict protein interaction given two input sequences. The results comparing this model to D-SCRIPT, presented in Table 3.2, are organized like those presented in Table 3.1 in the main text. Similarly to Chen et al.'s PIPR model, the Richoux et al. model performs better than D-SCRIPT when trained and tested on human PPIs while D-SCRIPT generalizes better across species, with substantially higher AUPRs than the Richoux model when the human-data-trained model is evaluated on other species. "D-HYBRID" and "P-HYBRID" refer to the same hybrid combination approach used for D-SCRIPT and PIPR.

Table 3.2: **Comparison with PPI prediction from Richoux et al.** We compare D-SCRIPT to the fully connected model described in Richoux et al. by a 5-fold cross validation in human, and the application of a human-data trained model to PPIs from 5 other species. As with the comparison to PIPR (see Table 3.1), while Richoux outperforms PIPR in the cross validation setting, D-SCRIPT generalizes significantly better to unseen proteins cross-species, and a hybrid model outperforms either model alone.

Species	Model	Precision	Recall	AUPR	AUROC
M. musculus	Richoux	0.749	0.321	0.518	**0.816**
	D-SCRIPT	0.818	0.346	0.580	0.833
	D-HYBRID	**0.821**	**0.354**	**0.608**	0.837
D. melanogaster	Richoux	0.582	0.126	0.231	0.659
	D-SCRIPT	0.798	0.359	0.552	0.824
	D-HYBRID	**0.799**	**0.360**	**0.559**	**0.824**
C. elegans	Richoux	0.672	0.079	0.252	0.671
	D-SCRIPT	0.840	0.306	0.548	0.813
	D-HYBRID	**0.841**	**0.307**	**0.554**	**0.814**
S. cerevisiae	Richoux	0.501	0.059	0.201	0.652
	D-SCRIPT	0.706	0.223	0.405	0.789
	D-HYBRID	**0.707**	**0.224**	**0.413**	**0.789**
E. coli	Richoux	0.746	0.093	0.303	0.687
	D-SCRIPT	0.791	0.520	0.571	0.863
	D-HYBRID	**0.792**	**0.391**	**0.581**	**0.863**
H. sapiens (5-fold cross validation)	Richoux	0.815	**0.737**	0.834	0.964
	D-SCRIPT	0.728	0.278	0.516	0.833
	P-HYBRID	**0.944**	0.405	**0.848**	**0.966**

3.4.3 D-SCRIPT predictions are functionally informative

The importance of PPI networks arises, in part, from the graph-theoretic analyses on them which enable the functional characterization of un-annotated proteins. We therefore sought to test if D-SCRIPT's success at cross-species generalization would translate to better functional inference in new species. In particular, we hypothesized that, compared to PIPR, the D-SCRIPT model trained on human data should facilitate more accurate inference of protein functional modules in *Drosophila melanogaster*. Towards this end, we generated a set of 10,475,595 candidate pairs from the set of *D. melanogaster* proteins in STRING. Using D-SCRIPT and PIPR's human-trained models, we predicted interactions over this candidate set. On the resulting PPI networks, we performed functional module detection and quantified

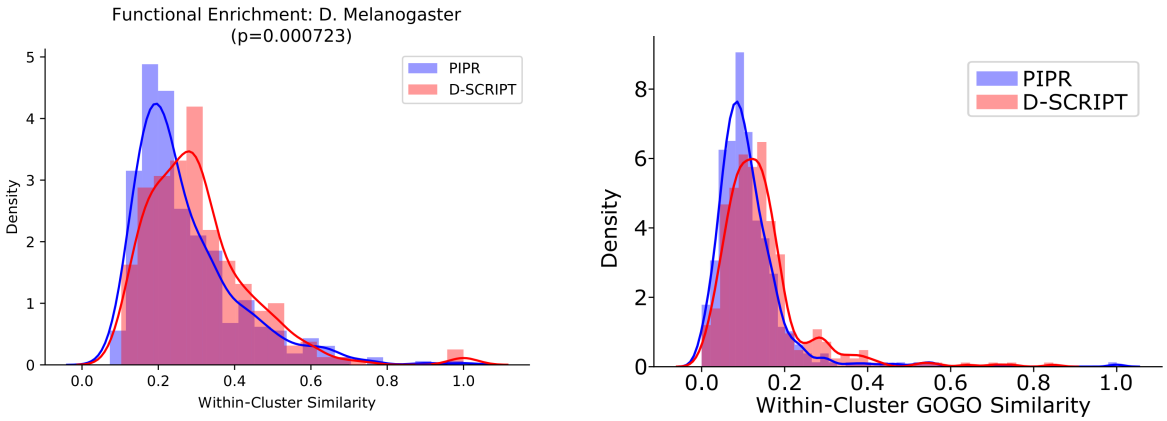


Figure 3.3: **Improved Protein Functional Characterization using D-SCRIPT Modules.** D-SCRIPT recovers more functionally coherent clusters than PIPR ($p = 0.000723$, one-tailed t-test). Clusters were generated by predicting protein interactions using either D-SCRIPT or PIPR, computing the diffusion state distance (DSD) between proteins, and clustering the DSD matrix using spectral clustering. Within-cluster similarity was calculated as the average Jaccard similarity between GO Slim annotations of all pairs of proteins in the cluster We also use Zhao et al.'s GOGO algorithm [61], to compute the similarity between pairs of GO terms (right); we then aggregate these to get within-cluster similarity scores. Similarity between proteins is calculated as GOGO Biological Processes similarity using all FlyBase GO term annotations. Our evaluation here yields similar results

the functional coherence of 374 (PIPR) and 384 (D-SCRIPT) modules using available GO (Gene Ontology) annotations from FlyBase [59] (Section 3.3). Functional coherence of a module quantifies the extent to which proteins in the module are likely to participate in the same biological functions. A higher average within-cluster similarity is desirable because it enables more accurate functional characterization of novel proteins by associativity and discovery of protein functional modules. We find that the average within-cluster similarity when interactions are predicted using D-SCRIPT is significantly higher than when using PIPR ($p = 0.000723$, one-tailed t-test), and that D-SCRIPT results in 24% more highly-enriched (top 10% of scores) clusters as PIPR (Figure 3.3). In addition to this cluster-based test, we directly compared the functional similarities between protein pairs (measured as the overlap of GO functional annotations) to their graph-theoretic similarities implied by the D-SCRIPT and PIPR networks (see Section 3.3 for details). We find that D-SCRIPT admits a significantly stronger correlation between the two measures than PIPR (Spearman

ρ, 0.123 vs. 0.005, $p = 0.00$, Fisher r-to-z). We additionally applied D-SCRIPT and PIPR to computationally screen all SARS-Cov-2 proteins against 19,777 human proteins, predicting approximately 3,000 viral-host PPIs from each method and characterized each viral protein's function by the GO annotations of its human interactors (Section 3.4.10). We find that compared to the corresponding annotations derived from 332 experimentally-determined PPIs (Gordon et al. [74]), D-SCRIPT based annotations overlap more with the experimental results than those from PIPR ($p = 0.059$, paired one-tailed t-test).

3.4.4 D-SCRIPT embeddings capture structure and interaction

One of our aims when designing D-SCRIPT was to capture the structural aspects of interaction — the per-protein embedding produced by the trained projection module should encode structural information. To examine this aspect, we randomly selected 300 proteins from the Protein Data Bank (PDB) and used $(n \times 100)$-dimensional D-SCRIPT embeddings of these proteins to predict protein structure. We randomly split the 300 PDB structures into a training set of 100 and a test set of 200, evaluating how well a logistic regression that uses the D-SCRIPT embeddings as the input predicts contacts between residues (Section 3.3). We show that a linear combination of features in the projection module output is able to recapitulate a meaningful subset of the ground-truth contacts, achieving a median per-structure AUPR of 0.19 over the test data set (Figure 3.4). These results strongly suggest that the end-to-end training of D-SCRIPT — using only sequence data — results in an intermediate representation that captures structural information at the level of each protein.

We also sought to benchmark the utility of D-SCRIPT's embeddings for predicting PPIs. We hypothesized that proteins which have similar embeddings are likely to interact in the same way, so it is possible to find new interacting pairs by searching for proteins which are similar to known interacting pairs. We compared D-SCRIPT embeddings with several other protein sequence representations: a one-hot embedding categorizing each amino acid into one of seven classes based on biochemical properties ("AAClass", from [79]), a 5-residue-context

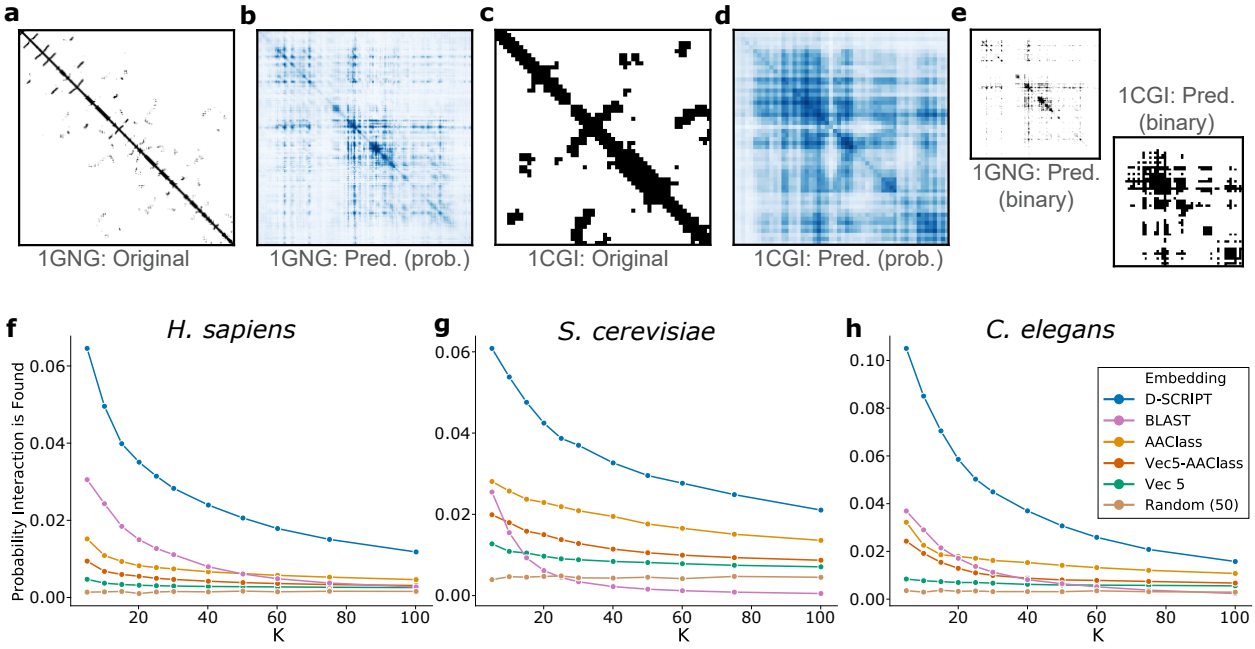


Figure 3.4: **D-SCRIPT Embeddings Represent Structure and Interaction** After a full model has been trained to predict interaction, the low dimensional embeddings learned by the projection module of D-SCRIPT can be used as meaningful representations of the protein in other applications. The PDB identifier 1GNG corresponds to a protein with 356 residues where the accuracy of using the D-SCRIPT embedding to predict self-contacts is near the median of cases we studied (AUPR=0.19), while 1CGI corresponds to a short protein (54 residues) in which the embedding achieves a higher accuracy (AUPR=0.38). The original contacts (**a, c**) were assessed at 8 Å. Contacts were predicted (**b, d**) using a logistic regression model trained on contacts seen in a held-out set of proteins, and the binarization thresholds for panel (**e**) were chosen so as to result in the same number of contacts as in the original maps. These embeddings also enable the accurate recovery of true interacting protein pairs in the neighborhood of known PPIs in human (**f**), yeast (**g**), and roundworm (**h**). D-SCRIPT embeddings recover more interacting proteins than any other embedding, regardless of species or number of neighbors checked. AAClass also performs well, likely because it characterizes biochemistry which is preserved at longer evolutionary distances. BLAST performs well at low values of k but has difficulty recovering interactions for larger values — likely due to network rewiring over longer evolutionary distances.

Skip-Gram embedding ("Vec5", from [80]), a concatenation of Vec5 and AAClass (used in the input for PIPR), and a randomly generated 50-dimensional embedding with values drawn uniformly from the range $[0, 1]$. Additionally, we used BLAST [19] to search for neighbors of interacting proteins. We then evaluated the number of interacting pairs we find in the neighborhood of a small set of "seed pairs", finding that D-SCRIPT finds more interactions in the nearest neighbors of the seed pairs than all other embeddings in *H. sapiens* (Figure

3.4f), *S. cerevisiae* (3.4g), and *C. elegans* (3.4h).

3.4.5 Predicted contact maps recapitulate known protein binding mechanisms

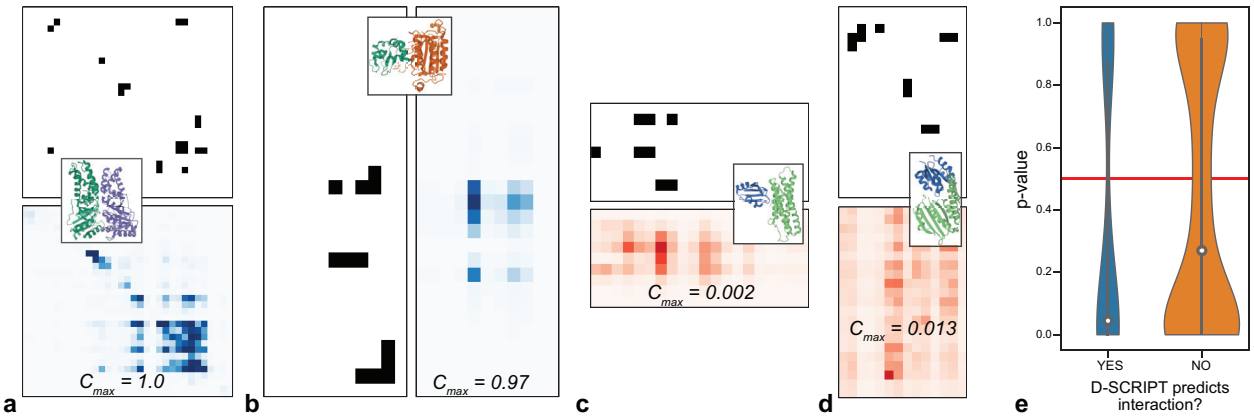


Figure 3.5: **D-SCRIPT Predicts Biologically Meaningful Contact Maps** We show inter-protein contact maps of protein structures known to dock together. Panels **(a,b)** correspond to pairs where D-SCRIPT correctly predicted an interaction, while panels **(c,d)** are cases where it incorrectly predicted no interaction. The black-and-white matrices correspond to the PDB ground truth while the colored matrices correspond to D-SCRIPT's $\hat{C}$; for the latter, the color scales of (a,b) differ from (c,d). While $\hat{C}$ contains large values for positive pairs, its maximum C_{max} is very low for negative pairs. Panel **(e)** shows a systematic evaluation of the 2-D Earth mover's distance-based similarity between $\hat{C}$ and the ground truth. Not only are the correctly-predicted $\hat{C}$s significantly similar to the ground truth, even when D-SCRIPT incorrectly predicts two proteins don't interact, its contact maps are still similar to ground truth.

We investigated whether the interpretability of our model could aid in predicting inter-protein docking contacts. As an intermediate representation, the D-SCRIPT Contact Module (Figure 3.2) produces an inter-protein contact map which predicts the probability of interaction between all pairs of residues in the candidate protein pair. We first sought to verify that the contact maps produced after training were consistent with our design goal: the maps corresponding to negative predictions should have uniformly near-zero contact probabilities, while those for positive predictions should be sparse but with isolated regions of high probability contact prediction. We found that this was indeed the case generally and show

some examples in Figure 3.5: the maximum predicted residue contact probability is high for positive examples and low for negative examples.

We next sought to test if the predicted contact map is physically representative of the actual docking mechanism of the interaction. We emphasize that this is a high bar given we do not provide any 3-D information to the model nor any guidance on docking and, in principle, the model could perform well on the classification task without a physically accurate contact map. We performed this test using Hwang et al.'s benchmark data set of docked protein structures [49]. For every pair of chains in each PDB complex in the benchmark set, we generated a candidate PPI. We applied our human-data-trained model on 295 candidate PPIs and evaluated the predicted contact maps against the ground-truth contacts (assessed at 8 Å).

In cases where our model predicted an interaction, we found the predicted contacts to indeed recapitulate the ground-truth contacts substantially (Figure 3.5a,b). Even in some of the cases where D-SCRIPT did not predict an interaction, the distribution of predicted contacts was nevertheless consistent with the ground-truth (Figure 3.5c). To systematically evaluate the accuracy of the D-SCRIPT contact map, we evaluated the distance between the predicted and true contacts using an optimal transport metric, and compared to a baseline established by randomly reshuffling the predicted matrix. We chose to measure similarity between regions of the two contact maps rather than measuring per-residue matches (with a metric such as binary cross-entropy) because the convolutional and max-pooling layers in our model aggregate over neighboring residues, thus diffusing the signal. We estimated the p-value of the predicted contacts against 500 random trials, finding that in cases where D-SCRIPT predicted an interaction, the contact-maps were substantially similar to the ground-truth (median FDR-corrected $q = 0.08$, one-sided t-test). Even in cases where D-SCRIPT did not predict an interaction, the similarity to the ground truth was higher than that of the random baselines (Figure 3.5e).

3.4.6 Using protein frequency counts as a predictor

PPI networks are dominated by a dense and highly connected core of nodes, where the core includes many hub nodes. A method that does relatively well on these hub proteins will have an advantage on within-species evaluations. For our cross-validation setup, we followed the precedent of previous work and split training and test data by edges [47], [50]. Consequently, a protein's relative frequency will be roughly similar in the training and test data sets. In this context, a PPI prediction heuristic that simply remembers frequency counts does strikingly well (see figure below). Here, we scored the likelihood of a candidate PPI (A, B) being true as the minimum of frequency counts, in training data, of A and B.

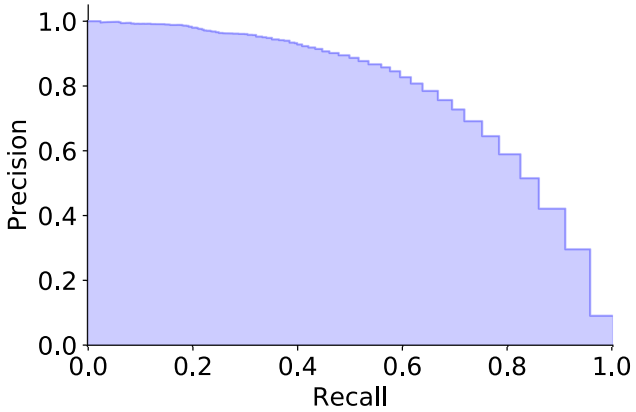


Figure 3.6: **Performance of a training-counts-only classifier.**: As part of the investigation of the impact of a protein's frequency in the training set on classifier performance, we created a naive classifier which simply predicts that a pair interacts with probability corresponding to the minimum number of times either of a pair exists in a positive interaction in the training set, normalized over the maximum number of times any such protein appears. We find that this naive classifier actually performs rather well in within-species cross-validation, achieving an AUPR of 0.784. However, such a classifier would output a probability of 0 for all interactions in a data set which did not share any proteins with the training set, as is the case in the cross-species setting.

3.4.7 D-SCRIPT performs better on infrequently occurring proteins

To further investigate the finding above, we ranked proteins in the human PPI network by their frequency of occurrence. For a set of quantiles $q \in [0, 1]$, we evaluated out-of-sample

D-SCRIPT and PIPR predictions on the human PPI sub-network consisting only of proteins of rank q or lower; here, lower q corresponds to a lower frequency of occurrence. In absolute terms, as the table below indicates, both D-SCRIPT and PIPR become more accurate at higher q. However, D-SCRIPT has a relative advantage at lower q (i.e., infrequently occuring proteins) while PIPR performs better at higher q. In other words, PIPR's better within-species performance can be traced to it being more accurate on proteins that occur frequently. This also suggests an explanation for PIPR's lower cross-species generalizability than D-SCRIPT: when making predictions on an entirely new set of proteins in a different species, knowing the relative frequencies of proteins in the training data might not be particularly useful.

The difference between D-SCRIPT and PIPR might stem from their respective architectures. The protein representation learned by D-SCRIPT is constrained to be a linear projection of the Bepler & Berger pre-trained embedding, albeit with ReLU and dropout layers. This regularizes how much frequency information can be incorporated into the model; we note that the Bepler and Berger model was trained with data on individual proteins and would not reflect PPI frequency information. In contrast, PIPR's design allows for a lot more leeway in training each protein's representation. This flexibility may allow PIPR to better incorporate the occurrence frequencies into its representation, helping its within-species performance but potentially hurting its cross-species generalizability.

Table 3.3: **Area under precision-recall curve (AUPR) of PPI prediction performance for interactions where proteins appear rarely in the training set.** We created increasingly restrictive subsets of the test data, where no protein in the subset may appear more than some cutoff number of times in the training set. While PIPR outperforms D-SCRIPT when frequently trained on proteins are present, at lower quantile values q, D-SCRIPT performs relatively better than PIPR - although the absolute performance of both classifiers drops off as the data set is more challenging.

Cutoff Quantile (q)	**Cutoff Value**	**% of Test Pairs**	**PIPR**	**D-SCRIPT**
1.0	209	100	**0.830503**	0.591353
0.9	12	75.31	**0.206161**	0.118704
0.8	5	61.12	0.061128	**0.068445**
0.7	2	46.85	0.014476	**0.024299**
0.6	1	38.03	0.005838	**0.011789**
0.5	0	25.06	0.001373	**0.006673**

3.4.8 Case study: Protein function and interaction in the bovine rumen

Because D-SCRIPT generalizes well to species with limited available PPI data, it enables the study of protein functional pathways through *de novo* prediction of protein interaction networks. Following this, we undertook a study of protein interaction in the bovine rumen to investigate the biological processes involved in rumination. In a comprehensive study of the sheep (*Ovis aries*) genome, Jiang et al. [81] identify several genes which are preferentially expressed in rumen tissue, including *PRD-SPRRII*, *S100-A2*, *S100-A12*, and *TCHHL2*. Using BLAST [19], we identify 12 putative homologous proteins in the ARS-UCD1.2 cow (*Bos taurus*) genome assembly (see Table A.2 for gene list) to focus on in our analysis. Including the rumen specific homologs, we selected 24,195 bovine proteins, and used the human-trained D-SCRIPT model to predict the probability of interaction for fifty million candidate pairs. We predicted a network of 476,399 positive interactions between 17,811 proteins, and performed functional module detection and gene set enrichment analysis (Section 3.3). To quantitatively assess the quality of our predicted edges and clusters, we computed the coexpression of each pair of genes (Figure 3.7f) in 93 tissue samples. We find that pairs of genes for which we predict an edge with D-SCRIPT are significantly more likely to be coexpressed than a random pair of genes ($p <$ 1e-84, one-sided t-test). Further, the correlation between expression vectors is even stronger for pairs which appear in the same functional module, even in cases where D-SCRIPT does not predict an interaction between the proteins.

The largest cluster we identify (Cluster A, Figure 3.7a) is comprised of 65 proteins, including 2 homologs of *PRD-SPRRII*. 34 of the genes in Cluster A are homologous to various human protein tyrosine phosphotases, across multiple classes of the protein tyrosine phosphotases categorized in [68], [69]. Other genes in the cluster are *TXNL1*, known to buffer response to oxidative stress [82], [83], and serine/threonine protein kinases including *VRK3* [84] and *STK33* [85]. Two genes in Cluster A have been previously been associated

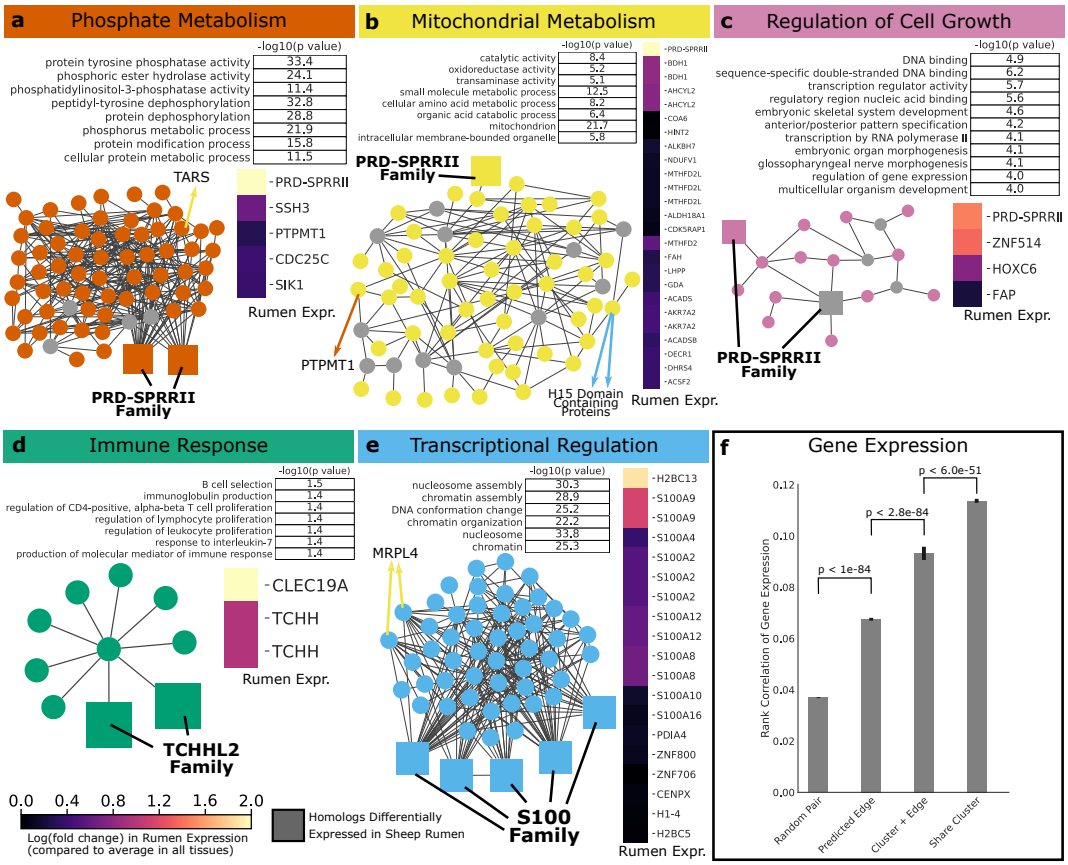


Figure 3.7: **Protein Interaction Network in Bovine Rumen** We apply D-SCRIPT to predict a *de novo* PPI network in cow (*B. taurus*), and apply spectral clustering to the diffusion state distance (DSD) matrix to identify functional modules, shortlisting five modules related to rumen physiology. A recent RNA-seq study validates several proteins in these modules as being strongly overexpressed in rumen tissue. For each module, we report gene ontology molecular function (GO:MF), biological process (GO:BP) and cellular compartment (GO:CC) annotations which are significantly enriched for the proteins in each cluster. We also show the log(fold change) for genes in the cluster which are more expressed in rumen tissue than on average across all tissues. For each module, nodes have been added in gray if necessary to fully connect all nodes. We find 3 modules containing members of the PRD-SPRRII family and which are enriched for phosphate and mitochondrial metabolism (**a,b**) and regulation of cell growth mechanisms (**c**). We also find a module with TCHH-like 2 proteins enriched for immune response (**d**), and with S100-A2 and S100-A2 proteins enriched for transcriptional regulation and chromatin organization (**e**). The modules in **a,b** and **e** are directly connected through *TARS* and *MRPL4*, which suggest a link between these functions in bovine rumen. In **f**, we demonstrate that protein pairs with a predicted D-SCRIPT edge correspond to a higher coexpression between their respective genes. This coexpression signal gets even stronger when evaluated only on protein pairs in a functional module, suggesting that both the protein network and functional modules are biologically meaningful.

with disease in cattle, *TUBD1* [86], where missense mutations were associated with increase juvenile mortality in cattle, and *NUAK1* [87], which was found to be differentially expressed in cows with milk fever.

This cluster is connected through *PTPMT1* and *TARS* to a cluster of 53 proteins (Cluster B, Figure 3.7b) which are active in metabolism within the mitochondria, and participate in oxidoreductase and transaminase catalytic activity, and which contains one homolog of *PRD-SPRRII*. Ruminants, like the cow, exhibit patterns of energy metabolism that are quite different from non-ruminants like humans or rodents [88]. Because ingested carbohydrates are fermented to short chain fatty acids in the rumen, glucose demand is met by gluconeogenesis, controlled by transcriptional regulation, and the associated genes and pathways are implicated in metabolic disorders that affect dairy cows such as fatty liver and ketosis. [89]. Many of whose human homologs of genes in Cluster B are known to localize to the mitochondria, including *ACSS1* [90], *AGXT2* [91], *COA6* [92], *DECR1* [93], *MTHFD2* [94], *OAT* [95], plus *LHPP* which was predicted by [96] to be involved in mitochondrial oxidative phosphorylation. Other genes in the cluster have been implicated in mitochondrial tRNA modification, including *CDK5RAP1* [97], *MTO1* [98], *TARS* [99] and *TFAM* [100]. Still others, such as *ABAT* [101], *FOXRED1* [102], and *NDUFV1* [103] have human homologs whose role in mitochondrial rare diseases has been documented. Several of the genes in this cluster have been implicated as important trait or disease markers in dairy cows, pigs, and sheep, making the predicted interactors of these genes of particular interest. For example, *BDH1*, involved in ketogenesis [104] has been linked by multiple studies to health of lactating dairy cows [104], [105]. High expression of *BDH1* was positively correlated with milk yield and negatively correlated with fat yield in buffalo [106]. Polymorphisms of the *DECR1* gene have been connected to meat quality [93], [107] and *AHCYL2* was one of ten candidate disease genes suggested by [108] to be involved in susceptibility to DA.

This cluster is further connected through *MRPL4* and two *H15* domain containing proteins to a module of 55 proteins (Cluster E, Figure 3.7e) involved in transcriptional regulation,

with significant enrichment for nucleosome and chromatin assembly and organization. In Wei et al. [109], *MRPL4* was identified as being involved in immune and inflammatory pathways, where the human homolog was suggested as a disease gene for allergic rhinitis. This cluster contains all four homologs of *S100-A12* and one of *S100A-2*, from the set of highly expressed sheep rumen protein homologs. It also contains homologs to other human S100 proteins, *S100-A4*, *S100-A7*, *S100-A8*, *S100-G*, *S100-A11* and *S100-A16*. Many of these S100 proteins have been implicated in progression of human epithelial tumor progression, cell differentiation, and chronic inflamation. [110]. A cluster of three of these proteins, S100-A12, S100-A7 and S100-A8, has been implicated in the innate immune response to pathogens, including parasites, *E. coli* and *H. pylori* [111], where some have been shown to function in the nutritional immunity mechanism, by out-competing bacterial metal ion transporters [2]. For the S100 family of proteins, however, there should be some caution in assuming specificity of function translates across species: for example, [112] showed that the Bovine S100-G most likely buffers calcium but is not likely to be a calcium sensor like mouse S100-G, despite over an 81% sequence identity. Still, we hypothesize a connection to anti-microbial activity and innate immunity for proteins in Cluster E.

We further identify two additional, smaller clusters. In one, the *PRD-SPRRII* homolog occurs in a module of 16 genes (Cluster C, Figure 3.7c) that also contains *HOXC6* and *HOXA7* and is enriched for multicellular organism development, skeletal system development, and organ morphogenesis, suggesting a role for *PRD-SPRRII* in cell growth in the rumen. Other genes in this cluster include *SNIP*, with anti-inflammatory function [113] and several genes whose human homologs *FAP*, *PRRX1*, and *TP73*, have been implicated in extra-cellular matrix remodelling, and cancer metastasis [114]–[116]. Both the *TCHHL2* homologs are part of the other small module (Cluster D, Figure 3.7d), which has 10 genes and is enriched for B cell selection and proliferation of CD4-positive alpha-beta T cells, lymphocytes, and leukocytes, suggesting that *TCHHL2* plays a role in the immune response within the bovine rumen. The *TCHHL2* protein may be involved in cross-linking keratins at the ruminal surface [117]. The

link between metabolism, cell growth and the immune system is documented in Turner et al. [118], and our analysis further suggests that these processes are involved in the modulation of immune response.

The above clusters contain many genes involved in rumen physiology and cow health, and with some of their homologs implicated in human diseases. However, like PPIs determined by *in vitro* assays, the predictions of D-SCRIPT should be cross-referenced with tissue-specific information when interpreting them in a particular tissue-type. Consider Cluster B, which consists predominantly of mitochondrial genes. Since D-SCRIPT has no knowledge of cellular compartments, this grouping has emerged naturally from the data, suggesting a biological signal. Indeed, we find many of these genes are highly expressed in the rumen (Figure 3.7b). On the other hand, 34 genes in Cluster A are protein tyrosine phosphatases (PTPs), all with similar but not identical functions. It is possible that only some of these genes are involved in rumen biology, with the rest active in other tissues. RNA-seq data supports this, identifying only a few PTPs as highly expressed in the rumen (Figure 3.7a). In general, the structure/function specificity of a protein family would help determine each member's tissue-specific selectivity. Interestingly, the large set of PTPs provides a natural setting to investigate D-SCRIPT's sensitivity to small sequence variations. PTP binding specificity is largely determined by the PTP catalytic signature motif (HCX_5R) [119]. In Section 3.4.9, we show that D-SCRIPT predicted probabilities of interaction between SPRR-II and PTP proteins drop substantially when the entire eight residue motif is perturbed, but remains high when only the flexible sites are randomized, indicating that D-SCRIPT is sensitive to residues which determine binding specificity. Further, we show that a systematic perturbation of each residue of the CDC14 subfamily of protein tyrosine phosphatases identifies the location of the catalytic signature motif completely *de novo*, which suggests that such an experiment could be used to form hypotheses about binding mechanisms of uncharacterized proteins.

3.4.9 Case study: Analysis of binding specificity in PTP family

Our analysis of functional modules in the bovine rumen found a single cluster containing 34 proteins homologous to human protein tyrosine phosphatases (PTPs). PTP's are known to comprise several families with similar sequence but diverse binding specificity, determined in part by a short catalytic signature motif [68], [69]. In Figure 3.8, we show Cluster A from Figure 3.7a, recolored based on the PTP sub-type. All but one of the neighbors of *PRD-SPRRII* in Cluster A are PTP proteins (the exception being *MARK2*, a serine/threonine-protein kinase). We find that D-SCRIPT does not seem to bind discriminately to one family, but tends to predict interactions across all sub-types.

To further investigate how D-SCRIPT determines binding specificity, we undertook an *in silico* mutagenesis experiment. The canonical catalytic signature motif for the PTP family is HCX_5R [119] or HCXXGXXR [69], a motif which we identified in 28 PTP proteins from Cluster A. We also included ENSBTAP00000067545 (*CDC25C*), which has the motif HCXXXXXA in our analysis. For each protein, we used D-SCRIPT to predict the probability of interaction with ENSBTAP00000070493 (*PRD-SPRRII*). Then for 50 replicates, we randomly perturbed the catalytic motif in that protein, either by randomly selecting amino acids for all 8 positions of the motif, or only the 5 flexible positions (X). We find that for 24 of the 29 proteins, perturbing only the flexible positions increases or does not change the D-SCRIPT predicted probability, while perturbing the entire motif drastically decreases the predicted probability. For 2 of the remaining 5, D-SCRIPT already did not predict an interaction with the original protein, for another one perturbing even the flexible positions decreased the probability of interaction significantly, and for the final 2 even perturbing the entire motif did not significantly decrease the predicted probability of interaction. Figure 3.9a shows the original prediction (black), the distribution of 50 replicates where only flexible sites were mutated (blue), and the distribution of 50 replicates where the entire catalytic motif was mutated (orange) for each PTP protein.

Finally, we sought to identify which residues were most important in determining the D-SCRIPT model's prediction. To do so, we selected 5 CDC14 proteins (ENSBTAP00000058782, ENSBTAP00000073534, ENSBTAP00000054725, ENSBTAP00000069880, ENSBTAP00000070948) and aligned them using MUSCLE [71]. Then, for each position in the alignment, for all sequences which didn't have a gap in that position, we randomly perturbed the amino acid at that position and used D-SCRIPT to predict interaction between the perturbed sequence and *PRD-SPRRII*. Figure 3.9b shows, for each position of the alignment, the difference between the average original predicted probability of interaction across the 5 sequences, and the average predicted probability of interaction across the perturbed sequences at that position. We find a very clear spike around the catalytic signature motif, indicating that D-SCRIPT is in fact basing its predictions on the residues involved in binding specificity. Further, when we zoom in to the 8-residue motif region, it is clear that D-SCRIPT is identifying the conserved part of the motif, and especially the 'C', as the most important residue in determining interaction. Figure 3.9c shows the WebLogo (https://weblogo.berkeley.edu/logo.cgi) for this region in the PTP domain, and the y-axis is the change in predicted probability when each position is perturbed.

3.4.10 Case study: Human interactions with SARS-CoV-2 proteins

We performed a preliminary study to predict viral-host interactions between SARS-CoV-2 and human proteins wherein we compared the sets of over-represented GO terms for human interactors of SARS-CoV-2 proteins, as predicted by D-SCRIPT or PIPR, with those over-represented in the experimentally-determined human interactors (Gordon et al. [74]). Figure 3.10 shows the relative similarity of computationally predicted annotations to the experimentally-determined annotations for each SARS-CoV-2 protein. Overall, we found that sets of enriched terms computed using the D-SCRIPT network overlap slightly more with the true network than those computed using the PIPR network ($p = 0.059$). Among the putative accessory factors (ORF* and Protein 14), D-SCRIPT performs significantly

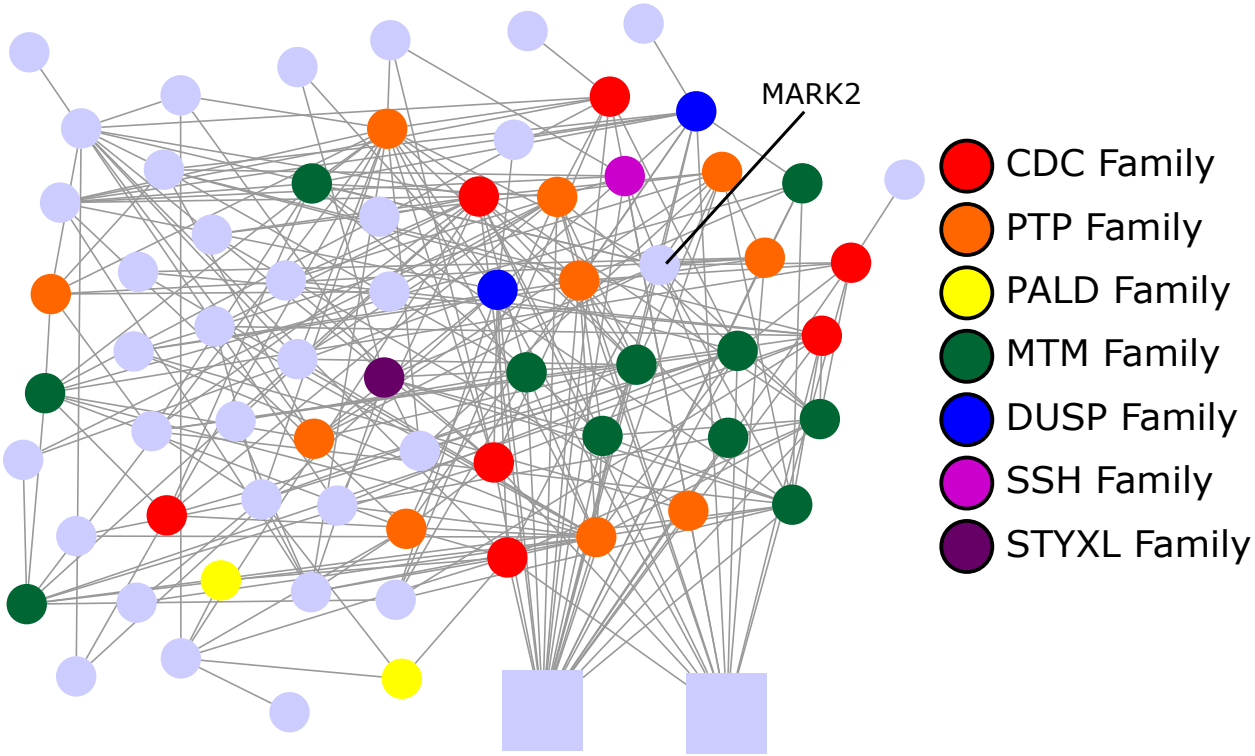


Figure 3.8: **PTP Subfamilies in Cluster A.** Cluster A (Figure 3.7a) colored by protein tyrosine phosphate (PTP) subfamily. All predicted interactors of the *PRD-SPRRII* family proteins (square) in Cluster A are homologs of human PTP proteins except for *MARK2*, a serine/threonine-protein kinase.

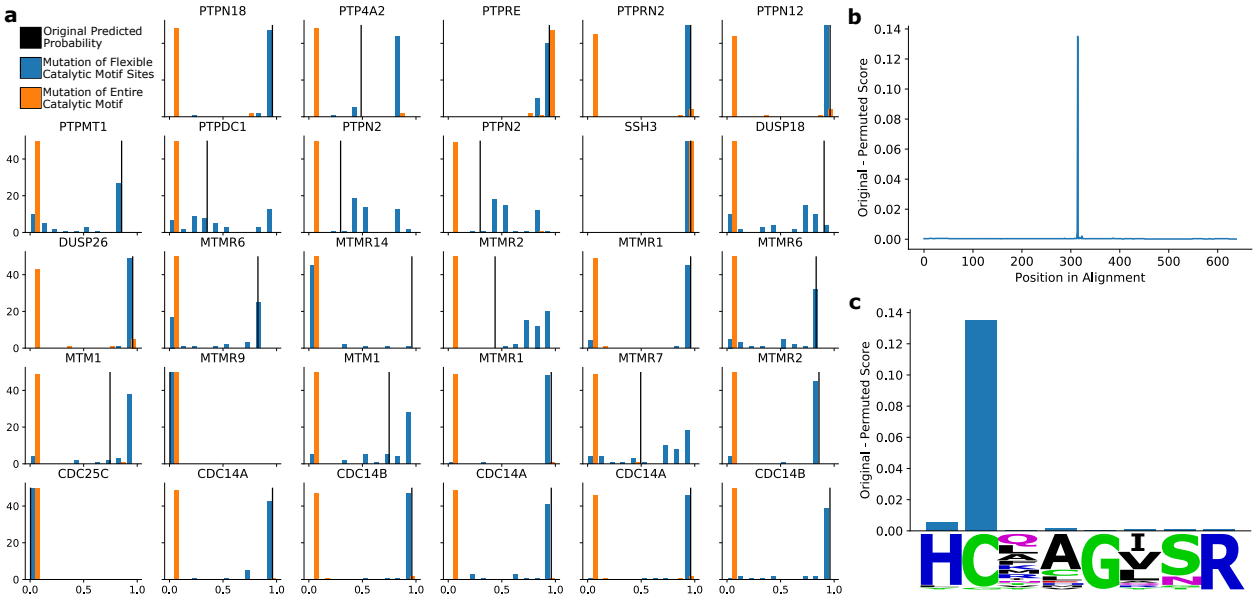


Figure 3.9: **D-SCRIPT Recognizes the PTP Catalytic Signature Motif.** **(a)** For each protein, we randomly perturbed only the flexible residues of the catalytic motif sequence, and the entire motif, 50 times. In blue, we show the distribution of D-SCRIPT predictions of interaction with PRD-SPRRII when only the flexible residues were perturbed. In orange, we show the distribution of predictions when the entire motif was occured. The black bar shows the original predicted interaction. In 24 of the 29 proteins, changing the flexible residues had little effect on the predicted probability, while changing the entire motif (including the non-flexible regions) significantly decreased the predicted probability. (**b**) For each position of the CDC14 alignment, we randomly perturbed each of 5 sequences at that position and predicted interaction using D-SCRIPT. Shown is the difference between the original and predicted probability when only that position is changed. The sharp spike around the 300 position identifies the catalytic signature motif. (**c**) The same experiment is shown, zoomed in to the 8-residue motif. D-SCRIPT clearly identifies the conserved 'C' position as most determinate in whether an interaction is predicted.

better (mean Jaccard similarity 0.029 vs. 0.118, $p = 0.022$, paired one-tailed t-test). Visually, PIPR seems to be somewhat better at predicting interaction partners for the non-structural proteins (NSP*), although D-SCRIPT still has a slightly larger mean similarity (0.183 vs. 0.222, $p = 0.221$). While D-SCRIPT performs better on the intensively studied spike (S) protein, PIPR shows a higher overlap for the nucleocapsid (N). Neither method predicts enriched terms for the other structural proteins encoding the envelope (E) and membrane (M) (0.149 vs. 0.121, $p = 0.672$ across the four proteins).

Candidate pairs were generated using the viral sequences from Gordon et al. [74] and 19,777 human sequences from the STRING database, and predicted edges using D-SCRIPT and PIPR. We predicted 3,273 edges using D-SCRIPT and 2,922 edges using PIPR. 332 putative true viral-host interactions were taken from Gordon et al. Human sequences were mapped to UniProt sequences identifiers from [74] with sequence similarity $\geq 95\%$ using BLAST [19], and UniProt identifiers were used to identify a set of Gene Ontology terms for the human interactors of each viral protein. Following [74], we identified over-represented GO terms using the clusterProfiler R package (version 3.14.3) [75] with a 1% false discovery rate (FDR). Over-represented GO terms were mapped to a common set of terms taken from the ChEMBL Drug Target GO Slim Subset [76]. For each viral protein, we computed the Jaccard similarity between the set of GO Slim terms enriched in the putative true network and each of the computationally predicted methods. We computed a paired one-tailed t-test to statistically compare the relative similarities of D-SCRIPT and PIPR.

3.4.11 Performance

D-SCRIPT took approximately 3 days to train for 10 epochs on 843,602 training pairs, and fits within a single 32GB GPU. Running time and GPU memory usage scales roughly quadratically, $O(mn)$, with the protein lengths m, n, since D-SCRIPT models the full $n \times m$ contact map as an intermediate step. Prediction of new candidate pairs with a trained model is very fast, requiring on average 0.02 seconds/pair and less than 5GB of GPU memory. Since

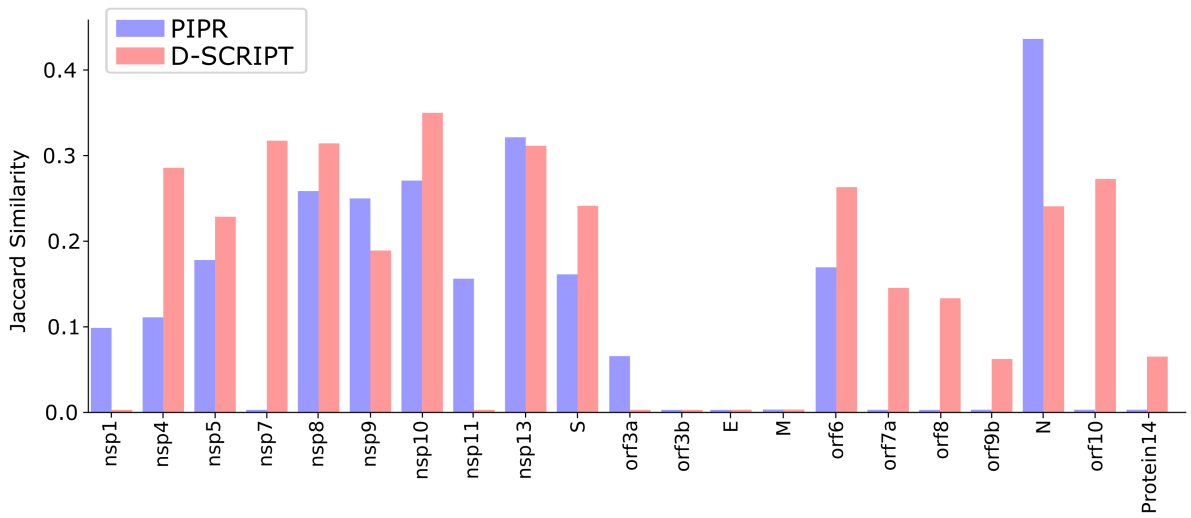


Figure 3.10: **Similarity of Enriched GO Terms to True Network.** Interactions between SARS-CoV-2 proteins and human proteins were predicted using D-SCRIPT and PIPR. Viral proteins were then annotated with the enriched ChEMBL GO Slim terms linked to their human interactors. Compared to PIPR, interactions computed with D-SCRIPT show a greater annotation overlap ($p = 0.059$) with those estimated from putative true interactions from Gordon et al. [74]. D-SCRIPT performs especially well at predicting functional enrichments for putative accessory factor proteins, where PIPR recovers none of the enriched functional terms in several cases.

D-SCRIPT generalizes well cross-species, it only needs to be trained once on a large corpus of data, and can be used to make predictions in a variety of settings.

3.5 Chapter Perspectives

We have introduced D-SCRIPT, an interpretable method for PPI prediction from sequence. D-SCRIPT takes a structure-based approach, with the prediction score for a protein pair computed as the binding compatibility of their respective structures. Since structure is more conserved than sequence over evolutionary time [120], this physical model of interaction generalizes well across species. Importantly, the intermediate contact map representation in the model is directly interpretable and can be used to validate the prediction or study the proteins' binding regions. D-SCRIPT thus joins the small but growing set of advances in interpretable deep learning methods in computational biology [121], [122]. Our modular

design additionally enables the investigation of model output at various stages, and we demonstrate that each layer captures incremental structural information.

The advantage of a sequence-based approach like D-SCRIPT is that the input sequence data is almost always available, due to the enormous advances in low-cost genome sequencing. Compared to PIPR [47], the state-of-the-art deep learning method that also takes sequences as inputs, D-SCRIPT generalizes better across species and can thus be more effective for accurate *de novo* PPI predictions in non-model organisms or less-studied proteins in organisms like fly. We suspect that D-SCRIPT's relative success across species, but under-performance on a within-species evaluation, is due to the simplicity of the model and the extent to which it is regularized. These design choices enhance D-SCRIPT's generalizability, directing it to learn general structural aspects of the interaction, rather than using network structure or the frequency of any individual protein as an interaction partner. However, for certain tasks a balance between the cross-species generalizability of D-SCRIPT and the within-species specificity of other state-of-the-art methods may be desirable. A future research direction might be transfer learning to tune a pre-trained D-SCRIPT model to a target species, while another approach could be to integrate it with guilt-by-association graph-theoretic PPI predictions [42].

Notably, D-SCRIPT does not require a multiple sequence alignment (MSA). Co-evolution based approaches that use MSAs have proven effective in reconstructing single-protein contact maps and 3-D structures [123]–[125]. When extending them to PPI prediction, an additional challenge is to identify the correct correspondence order between the rows of the two MSAs. In prokaryotic genomes where synteny conservation can be very informative, methods like ComplexContact [126], EV Complex [127], [128] and Gremlin [129] have been shown to perform well and provide residue-level interaction detail. However, there has been less success in extending these approaches to more complex, eukaryotic genomes. Importantly, we found the need to compute MSAs to be a performance bottleneck, making it infeasible to perform eukaryotic genome-scale predictions with them, therefore limiting the applicability of an

EV Complex approach in our setting. Nonetheless, incorporating co-evolutionary insights could improve D-SCRIPT's accuracy, and future work could explore ways to do so without sacrificing speed. Insights from related advances in the prediction of contact maps and structures of individual proteins could also be incorporated into our model architecture.

D-SCRIPT illustrates that learning the language of individual proteins, a successful deep learning effort, also helps decode the language of protein interactions. We leverage Bepler and Berger's pre-trained language model [48], [53], allowing us to indirectly benefit from the rich data on 3-D structures of individual proteins. In contrast, a PPI prediction method that was directly supervised with 3-D structures of protein complexes, in order to learn the physical mechanism of interaction, would need to contend with the relatively small size of that corpus [130]–[132].

There is a pressing need for scalable computational methods to infer a gene's function from its sequence in non-model organisms. While the sequencing revolution has helped make genomes more widely available, there remains a dearth of functional data. PPI prediction with D-SCRIPT is fast, making genome-scale screening feasible. For instance, we were able to evaluate 50 million candidate PPIs in *B. taurus* in 8 days on a single GPU. With D-SCRIPT, a workflow consisting of genome-scale PPI prediction, followed by graph-theoretic analysis of the PPI network to identify functional modules, can generate high-confidence predictions of gene function at scale; we demonstrated this in our cow rumen case-study. Such *de novo* PPI prediction can be useful even in model organisms such as *C. elegans*, for which the known portion of the PPI network is still quite sparse. In other organisms (e.g., *D. melanogaster*) where some PPI data does exist, future work could productively combine that data with D-SCRIPT predictions. We hope that its combination of broad applicability, cross-species accuracy, and speed will make D-SCRIPT a useful community resource for addressing the "genome to phenome" challenge.

Chapter 4

Protein-Protein Interactions II: Network and Structure

4.1 Chapter Overview

Computational methods to predict protein–protein interaction (PPI) typically segregate into sequence based 'bottom-up' methods that infer properties from the characteristics of the individual protein sequences (as we introduced in the previous chapter), or global 'top-down' methods that infer properties from the pattern of already known PPIs in the species of interest. However, a way to incorporate top-down insights into sequence-based bottom-up PPI prediction methods has been elusive. In this chapter, we introduce Topsy-Turvy, a method that newly synthesizes both views in a sequence-based, multi-scale, deeplearning model for PPI prediction. While Topsy-Turvy makes predictions using only sequence data, during the training phase it takes a transfer-learning approach by incorporating patterns from both global and molecular-level views of protein interaction.

A third type of method focuses on using structural, rather than sequence, information to make predictions of PPI. While high-quality structures have historically been limited, due to advances such as AlphaFold [133], high-quality computational structural models are

now pre-computed and available for nearly every protein in UniProt. However, the best way to leverage these models to predict which pairs of proteins interact in a high-throughput manner is not immediately clear. We also show here that using both the amino acid sequence and the 3Di sequence generated by Foldseek [134] as inputs to Topsy-Turvy substantially improves the performance of predicting protein–protein interactions cross-species. Thus TT3D (Topsy-Turvy 3D) presents a way to reuse all the computational effort going into producing high-quality structural models from sequence, while being sufficiently lightweight so that high-quality binary protein–protein interaction predictions across all protein pairs can still be made genome-wide.

4.2 Introduction

We focus on the problem of predicting PPIs from sequence data without the computational expense of multiple sequence alignments, thus enabling genome-scale predictions. Classically, the physical protein-protein interaction (PPI) prediction problem has been studied in two settings: one, where we only have access to each protein's amino acid sequence and must determine from the sequence data alone if the two proteins bind (e.g. [11], [47], [50], [135]). The other infers new interactions from the global topological properties of known PPI connections using either a simple rule such as "proteins with many common interaction partners are likely to also interact", or more sophisticated diffusion-based network embeddings (e.g. [37], [41], [42], [136]–[139]).

Our previous work introduced D-SCRIPT [11], a structure-aware deep-learning model for predicting protein interactions. D-SCRIPT takes a bottom-up view, learning about protein interactions pair-by-pair through the lens of (inferred) protein structure and, by leveraging a natural language based protein sequence representation, was shown to achieve state-of-the-art cross-species generalizability. While we originally trained D-SCRIPT on pairwise human PPI data, we pursue here the intuition that the wealth of network-level global information

available could potentially improve predictive performance if integrated during the training phase. Unfortunately, we found scant guidance in the literature for how to make use of both types of information simultaneously: existing PPI prediction methods (such as those listed above) either take exclusively a top-down or bottom-up approach, ignoring the other approach entirely.

Here, we propose a new approach, **Topsy-Turvy**, that integrates graph-theoretic (top-down) and sequence-based (bottom-up) approaches to PPI prediction in the training phase of our sequence-based predictor. Topsy-Turvy introduces a multi-objective training framework that takes a pair of protein sequences as input, with the supervision provided by *both* experimentally-determined PPIs (in the same manner as D-SCRIPT), as well as with global topological measures of protein pair compatibility. Importantly, it only requires protein sequences as inputs when making predictions— network information is used only during training. Since the trained Topsy-Turvy model makes predictions using just sequence data, it is particularly valuable in non-model organisms where almost no PPI data is available [11], [140]. We also investigate whether AlphaFold-Multimer [141], a very recent method for protein-complex structure prediction, can instead be adapted to solve our PPI prediction task; however, we found it to be 100,000 times slower than Topsy-Turvy. Due to its computational efficiency, Topsy-Turvy is applicable in genome-wide prediction settings where AlphaFold-Multimer would be infeasible.

While Topsy-Turvy requires no pre-existing experimental data in the species of interest, for cases where some such data *is* available (e.g., in worm or fly) we devise a hybrid model, **TT-Hybrid**, that is able to take advantage of species-specific network data. TT-Hybrid embodies a principled approach to combining the Topsy-Turvy sequence scores with GLIDE [42] scores to make PPI predictions; we chose GLIDE after benchmarking it against the widely-used node2vec [142] (Section 4.4.1). We show that TT-Hybrid performs better than its competitors, or just Topsy-Turvy or GLIDE alone.

This work has several key conceptual advances— (1) Whereas the D-SCRIPT algorithm

showed that informative features generated by a protein language model enable transfer learning of the structural basis of interaction, we show that we can likewise transfer global patterns of PPI organization by integrating a topological compatibility score into the loss function. (2) We approach the synthesis of bottom-up and top-down approaches as a multi-objective training problem that balances between structural and topological considerations when predicting PPIs. Except for the recent work of [143], such integrative approaches in prior work have been rare. (3) We provide a framework for accurately predicting PPIs in a variety of settings— both cross-species, where no training data is available in the target species, as well as in species that have limited experimentally-determined PPIs.

In a cross-species setting, Topsy-Turvy achieves state-of-the-art results, substantially improving upon the cross-species generalizability of PIPR [47], DeepPPI [78], and D-SCRIPT. We investigate Topsy-Turvy's improved performance, finding that it performs better not only on interactions involving hub nodes in the target species but even more so on low-degree nodes; this suggests that the measured outperformance is not simply due to ascertainment bias [144] (Sections 4.4.3, 4.4.4). We also investigated Topsy-Turvy's usefulness in settings where sufficient PPI data exists so that a putative interaction between two proteins *could* also be predicted using global methods. We show that TT-Hybrid's principled synthesis of the scores from the network-based GLIDE method [42] and Topsy-Turvy yields state-of-the-art performance in this setting as well.

Leveraging structural data when available Experimental PPI data remains sparse in most model organisms and even more so in other species. Recent deep learning methods that predict PPIs solely from sequence seek to address this limitation. We introduced D-SCRIPT [11] and Topsy-Turvy [12], two deep learning methods that rapidly predict whether two proteins will physically bind in the cell using only protein sequence information. We call these methods lightweight deep-learning methods, since they are computationally efficient enough to be run genome-wide. These methods can be contrasted with classical PPI docking

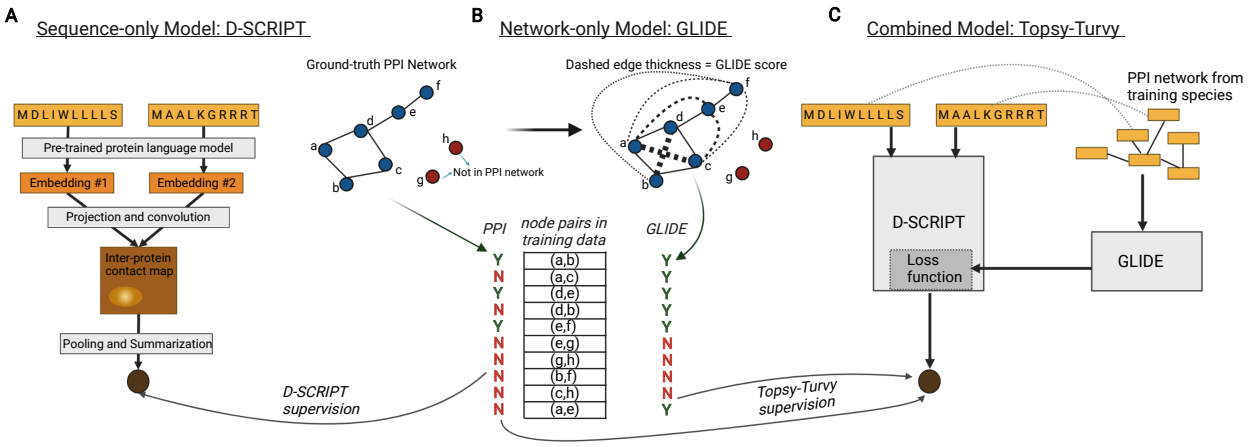


Figure 4.1: **Topsy-Turvy synthesizes sequence-to-structure based prediction using D-SCRIPT with network-based prediction using GLIDE. (A)** D-SCRIPT uses a protein language model to generate representative embeddings of protein sequences, which are combined with a convolutional neural network to predict protein interaction. It is supervised using binary interaction labels from the training network and regularized by a measure of contact map sparsity. **(B)** GLIDE scores all possible edges using a weighted combination of global and local network scores which are learned from the edges already in the training network. **(C)** Topsy-Turvy is supervised with both the binary interaction labels of the true (training) network and with the GLIDE predicted scores, thus integrating bottom-up and top-down approaches for PPI prediction into the learned Topsy-Turvy model.

methods [145] that require different inputs (namely the 3D structures of the proteins), and also produce different outputs (in addition to predicting *if* the proteins bind, they also model *how* they bind).

The advent of large deep learning methods for structure prediction like OmegaFold [146], AlphaFold2 [133], ESMFold [147] and RoseTTAFold [148], however, mean that high-quality 3D protein structural models can now be produced when only protein sequence is available as input. While these methods are too expensive to run from scratch at genome-wide scale, thanks to large community-wide efforts, there is no longer a need to run them from scratch: high quality computational structural models are now being made publicly available for nearly every protein in UniProt [149], [150]. In this work, we ask how this wealth of computational work and high-quality predicted structural information can be re-used to improve lightweight deep-learning methods that rapidly predict whether two proteins will physically bind in the cell. One potential approach is to run computational fold-and-dock methods such as

AlphaFold-Multimer [141], [151], or full complex structure prediction [152]–[155]. While these approaches are powerful for a small set of candidate pairs, they are still too computationally expensive to scale genome-wide, for example to create a full predicted PPI atlas for a non-model organism.

However, the wide availability of protein structure prediction methods has also coincided with breakthroughs in compact representation of protein structure and structure search. One such example is Foldseek [156], which uses a vector-quantized variational autoencoder (VQ-VAE) [157] to encode a protein structure as a sequence of discrete embedding vectors, each of which is then mapped onto a set of characters which called the 3D interaction alphabet (3Di). This process maps the three dimensional space of protein structure into a single dimensional 3Di sequence, which can then be used with fast sequence search tools such as BLAST [158] or MMseqs2 [159] to identify structurally similar proteins [160].

4.3 Methods

4.3.1 Overview of Topsy-Turvy

Topsy-Turvy provides a general paradigm to integrate a bottom-up sequence-based and top-down global method: for these two components in Topsy-Turvy we choose D-SCRIPT for the sequence-based prediction, and GLIDE for the network-base prediction. We next briefly review D-SCRIPT and GLIDE. In Topsy-Turvy, we adapt the D-SCRIPT model to synthesize the two by adding to it a network-dependent loss term inferred from the GLIDE model (Figure 4.1).

4.3.2 Data set generation

In order to select only high-confidence physical protein interactions, we limited our positive examples to binding interactions associated with a positive experimental-evidence score. From

this set, we removed PPIs involving very short proteins (shorter than 50 amino acids) and, due to GPU memory constraints, also excluded proteins longer than 800 amino acids. Next, we removed PPIs with high sequence redundancy to other PPIs Specifically, we clustered proteins at the 40% similarity threshold using CD-HIT , and a PPI (A-B) was considered sequence redundant (and excluded) if we had already selected another PPI (C-D) such that the protein pairs (A, C) and (B, D) each shared a CD-HIT cluster. Removing sequence redundant PPIs from the data set prevents the model from memorizing interactions based on sequence similarity alone [11].

4.3.3 Sequence-based prediction with D-SCRIPT

To make bottom-up, structure-aware predictions of PPIs, we use D-SCRIPT (introduced in Chapter 3), a state of the art method for sequence-based PPI prediction across species. Briefly, D-SCRIPT operates in two stages. First, we generate a feature-rich representation of each protein using a protein language model (PLM) by Bepler and Berger [48], [53]; next, these features are combined using a convolutional neural network to predict interaction. The Bepler & Berger PLM was chosen to extract structurally relevant features. Leveraging it, the D-SCRIPT architecture mimics the structural mechanism of protein interaction and includes an intermediate representation that encodes the intra-protein contact map. During inference, these predicted contact maps were shown to substantially recapitulate ground-truth binding mechanisms despite no structure-based supervision or inputs. To achieve this, the training procedure for D-SCRIPT minimizes a hybrid loss that contains terms measuring both the binary cross-entropy of predictions (L^{BCE}) and the overall magnitude of the contact map (L^{MAG}) which enables sparse and realistic contact map prediction. The relative weight of these loss terms are balanced by a hyperparameter λ. We emphasize that D-SCRIPT requires only the amino acid sequence of a protein pair to make predictions.

4.3.4 Network-based prediction with GLIDE

To make top-down, network-based predictions of PPIs in a species, we use GLIDE [42], a state-of-the-art method that combines local (neighborhood-based) and global (spectral) graph-theoretic techniques for quantifying the likelihood of an interaction between every protein-pair in the network. As part of our initial explorations, we also evaluated node2vec [142], another spectral approach for link prediction. However, we found GLIDE to outperform node2vec substantially on the PPI link prediction task (Section 4.4.1) and hence chose it as the link prediction technique in this paper. GLIDE combines a simple local score that captures shared-neighbor relationships in the dense core with a diffusion-based embedding that encapsulates the network structure in the periphery. While local metrics accurately capture the likelihood of links between proteins in the same local neighborhood, their performance drops significantly as the distance between proteins increases. The opposite is true for global metrics.

GLIDE incorporates both local and global metrics into a single score in such a way that each metric is leveraged in the region of the network where it is most accurate. We use Common Weighted Normalized (CWN) as our local metric, and the inverse of the Diffusion State Distance (UDSED$^{\gamma}$) as our global metric while computing the GLIDE score.

Following Devkota et al. [42], we compute the aggregate GLIDE score between each pair of nodes as:

$$GLIDE(p,q) = \exp\left(\frac{\alpha \cdot u(p,q)}{u(p,q)+\beta}\right) CWN(p,q) \quad + u(p,q) \tag{4.1}$$

where (p,q) is a candidate protein pair and $u(p,q) = 1/\text{UDSED}^{\gamma}(p,q)$. We chose the default values of α and β as suggested in [42] ($\alpha = 0.1, \beta = 1000$). These choices make the local embedding dominant, whenever available, with the global embedding being used to break ties and order nodes with the same local score. For the CWN local score, node-pairs with no common neighbors will have $CWN(p,q) = 0$ and only the global u term will be used.

4.3.5 Description of local and global similarity scores used in GLIDE

Local similarity score: Common Weighted Normalized Given nodes $p, q \in G$, the Common Weighted Normalized (CWN) score is

$$\text{CWN}(p,q) = \frac{\sum_{r \in \mathcal{N}_p \cap \mathcal{N}_q} (w_{p,r} + w_{q,r})}{\sqrt{k(p)k(q)}}$$

where for any node $x \in G$, $\mathcal{N}_x$ is the neighbor set of x, $w_{x,y}$ is the weight of the edge (x, y) and $k(x)$ represents the weighted degree of x. Note that this is slightly different from the CW metric described in [42], because of the square roots in the denominator, which we found corrected an overweight on the interactions between high-degree hub nodes from the original CWN used in GLIDE, improving performance.

Global similarity score: UDSED$^\gamma$ Distance We first describe the DSE$^\gamma$ embedding that forms the basis of this scoring scheme (from [42]). Let P be the Markov transition matrix computed from a graph G with the unique stationary distribution π and let D be the diagonal degree matrix representing the weighted degree of all the nodes in the network. Then the DSE$^\gamma$ embedding is:

$$DSE^\gamma = I + \sum_{t=1}^{\infty} \gamma^t (P - W)^t, \tag{4.2}$$

where W is a constant matrix, whose rows are copies of the stationary distribution π and γ is a parameter satisfying $0 < \gamma \leq 1$, which is used to control the contribution of larger time-steps in the computation of the embedding. We set $\gamma = 1$ in all our experiments, as suggested in [42].

If DSE$^\gamma(p)$ and DSE$^\gamma(q)$ represent the DSE$^\gamma$ embeddings for the nodes p and q respectively, we consider the un-normalized L2 distance between their DSE$^\gamma$ embeddings. Again, this is a

variation from *normalized* L2 distance described in [42]. Formally, this can be written as

$$\text{UDSED}^{\gamma}(p,q) = \sqrt{\sum_{k}(\text{DSE}^{\gamma}(p)_k - \text{DSE}^{\gamma}(q)_k)^2} \tag{4.3}$$

4.3.6 Network-dependent loss term

Topsy-Turvy retains the protein language model feature generation and convolutional neural net architecture of D-SCRIPT, with changes made to the training approach and loss function. To synthesize this model with link-based prediction, we introduce the additional task of predicting GLIDE scores between proteins, formulating it as an extra loss term in the objective. The entire model is then trained end-to-end.

In the original D-SCRIPT model, the loss function was a weighted sum, $L = \lambda L^{BCE} + (1-\lambda)L^{MAG}$, that combined the binary cross-entropy (BCE, [11]) loss with a regularization penalty related to the contact map's magnitude. To incorporate a network term, we add a sub-objective to the classification component:

$$L = \lambda(L^{BCE} + g_p L^{GLIDE}) + (1-\lambda)L^{MAG} \tag{4.4}$$

where L^{GLIDE} represents the loss when predicting GLIDE estimates and $0 \leq g_p \leq 1$ is a hyperparameter indicating its relative importance (at $g_p = 0$, the function reduces to the original D-SCRIPT loss). To compute L^{GLIDE}, we first generate GLIDE scores for every negative training example by computing the component CWN and UDSED^{γ} scores on the PPI network defined by the positive examples in the training set. For a protein pair (p, q), the loss L^{GLIDE} is defined as

$$L^{GLIDE}(p,q;\ g_t) = \text{BCE}(\ y(p,q), \mathbb{1}_{GLIDE(p,q) \geq g_t}) \tag{4.5}$$

where $g_t > 0$ is a hyperparameter, $y(p, q)$ is Topsy-Turvy's predicted score for the protein pair (p, q). $\mathbb{1}$ is the indicator function corresponding to the predicate $GLIDE(p, q) \geq g_t$. This formulation corresponds to binarizing GLIDE scores at the score threshold g_t and then applying the standard BCE loss. For convenience, we define g_t in terms of a percentile cutoff on the distribution of $GLIDE(p, q)$ scores (i.e., $0 < g_t < 100$), rather than directly as a numeric threshold.

In formulating L^{GLIDE}, we chose to binarize GLIDE scores and compute a BCE loss, rather than keeping continuous-valued GLIDE scores and using a different functional form for the loss. Doing so allowed us to mimic the form of the existing BCE-based loss, letting us calibrate the relative weights of L^{BCE} and L^{GLIDE} simply by g_p. Using GLIDE's continuous scores would have made this calibration difficult, since the un-normalized GLIDE scores are unevely distributed (for the human PPI training network: minimum = 0, median = 0.31; 75th-percentile = 0.40; maximum = 2.71) and do not follow a convenient closed form.

The addition of the GLIDE loss term to the model training accounts for the observation that the original D-SCRIPT loss measures only pairwise interaction, and is unaware of global network structure. Since the GLIDE score of a protein pair takes into account local and global network properties, the GLIDE component of the loss should incorporate network-wide information into the predictions. Specifically, since D-SCRIPT prioritizes precision and is more likely to miss true interacting pairs than GLIDE, the absence of strong structural evidence of interaction could be supplemented by strong network evidence.

4.3.7 TT-Hybrid can use a known network during inference

During inference, Topsy-Turvy requires only protein sequences as input. When making predictions in a species where some PPI data is also available, predictions from pre-trained Topsy-Turvy (trained on data from another species) can be combined with GLIDE predictions informed by the target species' PPI network. We note that these GLIDE scores are distinct from those corresponding to the training species; the latter were used only during training.

To take advantage of the PPI network in the target species when available, we designed TT-Hybrid that can be applied on query protein-pairs where both GLIDE and Topsy-Turvy scores are available. We note that this requires both proteins of the queried pair to be present in the target species' PPI network; otherwise, only Topsy-Turvy can be used. TT-Hybrid computes a weighted sum of Topsy-Turvy and GLIDE predictions for a query protein-pair, with the score for a protein pair (p, q) being:

$$\text{TT-Hybrid}(p, q) = 1 \cdot GLIDE(p, q) + w \cdot \text{Topsy-Turvy}(p, q) \tag{4.6}$$

For simplicity, we have set the weight of GLIDE scores to 1, since only the relative weighting of the two scores matters. In this paper, we trained Topsy-Turvy on human PPI data and have evaluated it on other species. During the training phase, we held out some human PPI data for validation. We calibrated w on this held-out human data using logistic regression.

We started by selecting protein pairs corresponding to the edges of the held-out human PPI subnetwork (see Section 4.4.2 for dataset details). These pairs were labeled positive; negatively-labeled pairs corresponded to random pairs of proteins from the subnetwork. The ratio of negative to positive examples was set to 10:1 to account for the inherent class imbalance in PPI data (see Section 4.4.2 for discussion). To avoid bias arising from data leakage, we also required that none of the examples occur in the original training data for Topsy-Turvy. We computed GLIDE and Topsy-Turvy scores for each protein pair, these methods having previously been trained on the rest of human PPI data. We then fitted a logistic regression model that sought to predict the label of a protein pair using its GLIDE and Topsy-Turvy score. The TT-Hybrid calibration weight w is chosen as the ratio of logistic regression coefficients, $c_{\text{Topsy-Turvy}}/c_{\text{GLIDE}}$. Our computation yielded $w = 0.3268$, and we recommend the use of this value when applying TT-Hybrid in other species, as is done in the

results presented here. If enough PPI data is available in the target species that a portion of it can be set aside, the held-out portion can be used to calibrate w specifically for the target species. To avoid the risk of data leakage, however, the same set of PPIs should not be used to both calibrate w and compute the GLIDE score inputs to TT-Hybrid.

4.3.8 Hyperparameter selection and model training

The hyperparameters g_p (the relative weight of GLIDE vs. binary cross-entropy loss) and g_t (the binarization threshold for GLIDE scores) play a crucial role in Topsy-Turvy and we sought to estimate them from cross-validation runs on the human PPI dataset. We note that all Topsy-Turvy and TT-Hybrid results presented in this paper are from models trained on human data but evaluated on out-of-sample, non-human data. To perform the hyperparameter search, we did cross-validation runs on the *entire* human PPI network, since GLIDE scores computed on smaller subnetworks might not be representative of the full network's characteristics. Due to the computational expense of such runs, however, we modified the standard grid-search approach. Initial, small scale explorations suggested $g_t = 90$ to be a promising choice. We first performed a grid search on g_p, fixing g_t to 90. This yielded $g_p = 0.2$ as the suggested choice (Table 4.1a) and we then performed a grid search for g_t, with g_p fixed to this choice. The second search indicated $g_t = 92.5$ to be the best choice (Table 4.1b), and we accordingly chose $g_p = 0.2$, $g_t = 92.5$ as the hyperparameter settings for Topsy-Turvy training.

4.3.9 Integrating structure information with TT3D

TT3D augments the inputs to the basic Topsy-Turvy architecture with encodings of the Foldseek-generated 3Di sequence (see Figure 4.2) [156]. In Topsy-Turvy, the amino acid sequence $x = x_1x_2...x_n$ is numerically encoded using the Bepler & Berger protein language model [48], [53] as $X \in \mathbb{R}^{n\times 6165}$, which is then reduced in dimension via a multi-layer perceptron to a projection $X^* \in \mathbb{R}^{n\times 100}$.

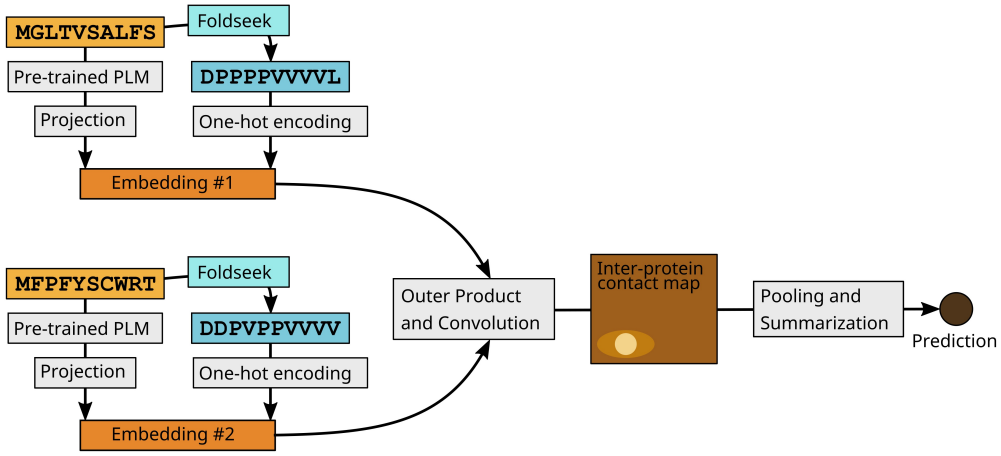


Figure 4.2: **TT3D model architecture.** TT3D follows the structure and training procedures of Topsy-Turvy, but with an augmented protein embedding. We concatenate a one-hot encoding of the Foldseek 3Di [156] string to the protein language model (PLM) based embedding before passing this representation into the convolutional portion of the architecture.

In TT3D, we additionally convert the protein sequence x to a 3Di sequence $y = y_1 y_2 ... y_2$ using Foldseek. If a crystal structure is available for the protein, Foldseek can be directly applied. If the sequence is not available in the PDB, we query for an exact match for it in AlphaFoldDB [149], and this structure is then used for extraction of the 3Di sequence by Foldseek. If no such hit can be found, we conservatively add an uninformative all-X 3Di sequence. We represent y with a one-hot encoding, yielding $Y \in \mathbb{R}^{n \times 21}$. We then concatenate the embeddings from the language model and from Foldseek, resulting in a joint embedding $E = [X; Y] \in \mathbb{R}^{n \times 121}$. Given two protein sequences x_1, x_2, we combine embeddings E_1, E_2 as in D-SCRIPT and Topsy-Turvy [11], [12] to predict a probability of interaction. The Topsy-Turvy loss function is used to train the model using back-propagation.

4.3.10 Availability and Implementation

Topsy-Turvy We implemented Topsy-Turvy in PyTorch 1.2.0 and trained with a NVIDIA Tesla V100 with 32GB of memory. Embeddings from the pre-trained Bepler and Berger model were produced by concatenating the final values of the output and all hidden layers. Apart from these pre-trained embeddings, Topsy-Turvy was trained end-to-end and did not

use pre-trained D-SCRIPT model weights. However, we used the same hyperparameters as in [11] for the relevant components of our model's architecture: a projection dimension of $d = 100$, a hidden dimension of $h = 50$, a convolutional filter with width $2w + 1 = 7$, and a local max-pooling width of $l = 9$. Furthermore, we used $\lambda = 0.05$ for calculating the training loss, choosing it based on early, small-scale explorations. Weights were initialized using PyTorch defaults. Model training parameters were set within ranges commonly used in deep learning literature: we used a batch size of 25, the Adam optimizer with a learning rate of 0.001, and trained all models for 10 epochs.

Table 4.1: **Hyperparameter search**: cross-validation AUPR (area under precision-recall curve) scores on full human PPI network for a) grid search for g_p, with g_t fixed to 90 (estimated from small-scale explorations), b) grid search for g_t, with g_p fixed to 0.2 (i.e., the optimal value from (a)). The metrics reported in the tables are the validation AUPR scores maximized over three epochs of training.

(a) At $g_t = 90$

g_p	AUPR
0.1	0.739
0.2	**0.802**
0.4	0.759
0.8	0.760

(b) At $g_p = 0.2$

g_t	AUPR
90	0.697
92.5	**0.824**
95	0.691
97.5	0.690

Topsy-Turvy 3D For inference with TT3D, as well as with Topsy-Turvy and D-SCRIPT, we make available a web interface at https://cb.csail.mit.edu/cb/dscript/. This interface is implemented with Gradio [161] and hosted on HuggingFace spaces, and allows the user to upload a `.fasta` formatted file with sequences and a `.tsv` file with candidate protein pairs, and get back predictions for the desired model. This interface additionally leverages 3Di sequences from [162].

For model training or larger-scale inference from the command line, TT3D is implemented in Python 3 as part of the `dscript` package for predicting protein-protein interactions, which is available from the PIP package repository (`pip install dscript`) or on GitHub at https://github.com/samsledje/D-SCRIPT. Model training and inference was performed on

a machine with a 112-core Intel Xeon Gold 6258R CPU and using a single NVIDIA A100 GPU. TT3D is trained for a maximum of 10 epochs, and the best performing model in cross-validation is used for making predictions. We make the trained model available to download at https://d-script.readthedocs.io/en/stable/, where it can be used to make new predictions with the `dscript predict` command.

TT3D requires that Foldseek [156] be installed and that 3Di sequences be generated for protein sequences in the training or inference set. Structures in `.pdb` format must be available for all sequences, either natively or generated by a structure-prediction method such as OmegaFold [146], AlphaFold2 [133], or RoseTTAFold [148]. Foldseek can be downloaded and build from source on Github at https://github.com/steineggerlab/foldseek. For convenience, we provide the command `dscript extract-3Di`, which uses the user's installed Foldseek to translate a set of structures into a `.fasta` file containing 3Di sequences.

To run TT3D, users should run the command `dscript train -allow_foldseek`, where `-allow_foldseek` is an optional command that runs the training iterations in "Foldseek" mode. While running in this mode, the user should provide the corresponding 3Di sequences in `.fasta` format using the `-foldseek_fasta` argument.

4.4 Results

We start by presenting a comparative assessment of GLIDE and node2vec for PPI link prediction; the results of this analysis motivated our choice of GLIDE as the network-theoretic component of the Topsy-Turvy model. We next evaluate the cross-species generalizability of Topsy-Turvy, showing how incorporating network data during training results in superior performance in other species, using only sequence data for prediction. We note that in the typical cross-species setting, purely network-based methods like GLIDE are not applicable since they can only make predictions for pairs where both proteins exist in the training PPI network and hence can not be applied to out-of-sample proteins. We therefore evaluated

Topsy-Turvy against methods that require only sequence-based inputs (like D-SCRIPT), assessing if co-supervising Topsy-Turvy with topological information allows it to learn aspects of protein interaction that carry across species. As we show, it does, and in subsequent analyses we investigate various aspects of the comparison more deeply, also addressing the issue of ascertainment bias in the evaluation network. Lastly, we study how to best apply Topsy-Turvy in instances where PPI data *is* available and GLIDE would be applicable directly. We find that while GLIDE is broadly informative about the species-specific network rewiring, better performance can be achieved by TT-Hybrid, a combination of Topsy-Turvy and GLIDE.

4.4.1 Comparison of GLIDE and node2vec

In our initial explorations, we sought to identify the most appropriate top-down PPI link prediction technique. Towards this, we compared GLIDE to node2vec [142]. The node2vec algorithm, also a spectral approach, uses a biased random walk procedure to construct low-dimensional node embeddings. Following the original study, we trained a logistic regression classifier on the Hadamard product of the node embeddings to predict the existence of a link given two candidate proteins. We compared the two methods on the *Drosophila* BioGRID network consisting of 3,093 nodes and 25,427 edges. A certain fraction $1-p$ of the edges were removed from the network (while protecting a random spanning tree to ensure connectivity), and the remaining subnetwork was used to train the node2vec and the GLIDE models. The removed edges were then used as positive test examples for evaluation. For negative examples, we randomly sampled 254,270 node-pairs (or 10 times the positive edge count) that were not present in the original network. The negative examples, like the positive edges, were also separated into train and test sets using the same parameter p. The dimension of the node2vec embedding was set to 300, i.e., approximately 10% of the node count (following Cho et al. [36]; this is also higher than the minimum value of 100, as prescribed by Grover et. al.). We evaluated both node2vec and GLIDE for different values of p (which correspond to

varying levels of network sparsity), finding that GLIDE outperformed node2vec consistently (Table 4.2).

Table 4.2: **GLIDE and node2vec comparison**: AUPR scores for PPI prediction on the *Drosophila* BioGRID network. Higher values of p correspond to a higher proportion of edges preserved in the training network.

p	GLIDE	node2vec
0.8	**0.737**	0.681
0.6	**0.818**	0.721
0.4	**0.839**	0.664
0.2	**0.805**	0.574

Table 4.3: **Topsy-Turvy improves upon D-SCRIPT [11], PIPR [47], and DeepPPI [78] for cross-species PPI prediction.** All species were evaluated using models trained on a large corpus of human PPIs. For D-SCRIPT and Topsy-Turvy, we report the average and standard deviation of results from three random initializations. For PIPR and DeepPPI, we report here the results from the study in [11] where the same evaluation scheme and data was used. For all data sets, there is a 1:10 ratio of positive to negative pairs, which means a random baseline would have an AUPR of 0.091 and an AUROC of 0.5.

Species	**Model**	**AUPR**	**AUROC**	**FPR**	
				0.1 Recall	**0.5 Recall**
M. musculus	PIPR	0.526	0.839	0.002	0.057
	DeepPPI	0.518	0.816	**0.0002**	0.059
	D-SCRIPT	0.663 ± 0.05	0.901 ± 0.02	0.002	0.014
	Topsy-Turvy	$\mathbf{0.735 \pm 0.03}$	$\mathbf{0.934 \pm 0.01}$	0.001	**0.009**
D. melanogaster	PIPR	0.278	0.728	0.007	0.197
	DeepPPI	0.231	0.659	0.012	0.274
	D-SCRIPT	0.605 ± 0.06	0.890 ± 0.02	0.003	0.022
	Topsy-Turvy	$\mathbf{0.713 \pm 0.05}$	$\mathbf{0.921 \pm 0.02}$	**0.001**	**0.011**
C. elegans	PIPR	0.346	0.757	0.002	0.148
	DeepPPI	0.252	0.671	0.007	0.252
	D-SCRIPT	0.550 ± 0.08	0.853 ± 0.04	0.003	0.032
	Topsy-Turvy	$\mathbf{0.700 \pm 0.04}$	$\mathbf{0.906 \pm 0.03}$	**0.001**	**0.011**
S. cerevisiae	PIPR	0.230	0.718	0.017	0.213
	DeepPPI	0.201	0.652	0.018	0.288
	D-SCRIPT	0.399 ± 0.09	0.790 ± 0.06	0.005	0.089
	Topsy-Turvy	$\mathbf{0.534 \pm 0.01}$	$\mathbf{0.850 \pm 0.02}$	**0.002**	**0.038**
E. coli	PIPR	0.271	0.675	0.005	0.246
	DeepPPI	0.271	0.688	0.004	0.243
	D-SCRIPT	0.513 ± 0.09	0.770 ± 0.03	0.002	0.040
	Topsy-Turvy	$\mathbf{0.556 \pm 0.09}$	$\mathbf{0.805 \pm 0.07}$	**0.001**	**0.038**

4.4.2 Performance is improved by integrating topology information from the training network

Datasets We trained Topsy-Turvy on human PPI data and evaluated it on *M. musculus*, *D. melanogaster*, *C. elegans*, *S. cerevisiae*, and *E. coli*. The data set selection and pre-processing follows [11]: we sourced positive examples from the STRING database (v11) [163], selecting only physical binding interactions associated with a positive experimental-evidence score. Our human PPI set consists of 47,932 positive and 479,320 negative protein interactions, of which we set apart 80% (38,345) for training and 20% (9,587) for validation (see Section 4.3.2 for details). For each of 5 model organisms (Table 4.3) we selected 5,000 positive interactions and 50,000 negative interactions using this procedure, with the exception of *E. coli* (2,000/20,000) where the available set of positive examples in STRING was limited. Each model was trained three times, with different random seeds, and we evaluated the average performance across these runs. We emphasize that Topsy-Turvy is trained end-to-end and does not use a pretrained D-SCRIPT sub-component. For benchmarking, a separate D-SCRIPT model was trained and evaluated identically.

In Table 4.3, we report the area under precision recall curve (AUPR) and area under receiver operating curve (AUROC) for each model in each species. As our dataset and evaluation approach is the same as in [11], we also include results reported there for two other state-of-the-art sequence-based PPI prediction methods, PIPR [47] and DeepPPI [78]. We note that for unbalanced data, AUPR is generally considered the more representative metric. We also report the false positive rate (FPR) at 10% and 50% recall, which measures the likelihood that a protein pair predicted to interact is incorrectly classified — an important metric in the case where high-likelihood pairs are then tested experimentally. We find that Topsy-Turvy achieves the highest AUPR and AUROC of all the methods we evaluated in each of five species, and has the lowest FPR at both recall levels. We also observe that Topsy-Turvy retains the structural interpretability of D-SCRIPT: for each queried protein

pair, the model also outputs a predicted inter-protein contact map for the putative binding between the two proteins.

Runtime and memory usage Topsy-Turvy took approximately 79 hours to train for 10 epochs on 421,792 training pairs, and fits within a single 32GB GPU. Running time and GPU memory usage, like in D-SCRIPT, scales quadratically, $\mathcal{O}(nm)$, with protein lengths n, m, since Topsy-Turvy models the full $n \times m$ contact map as an intermediate step. The prediction of new candidate pairs with a trained model is very fast, requiring on average 0.02 s/pair. Since Topsy-Turvy generalizes well across species, it needs to be trained only once on a large corpus of data and can be used to make predictions in a variety of settings. The additional run time for TT-Hybrid is minimal (approx. 15 minutes, most of it for GLIDE) since it just computes a weighted sum of predictions from Topsy-Turvy and GLIDE. The actual computation of TT-Hybrid scores, provided that the Topsy-Turvy and GLIDE results are already available, is a linear time operation (less than 1 minutes for the candidate set with 10 million pairs) since it is simply a weighted sum of the two.

Using network-level information for negative edge selection

Notably, Topsy-Turvy achieves greater cross-species generalization even though network information is used only during training. We hypothesize this may be partially due to GLIDE-based interaction scores mitigating the impact of incorrect labels in training data. To create negative training examples, we followed the common practice of randomly selecting protein pairs not experimentally reported as interacting [11], [47], [50]. However, it might be that such a pair actually *does* interact but has not yet been experimentally assayed. In such cases, the GLIDE score for the pair is likely to be high, thus improving the supervision and training of Topsy-Turvy. To further investigate our hypothesis, we evaluated an alternative approach to incorporating network topology in the model, by modifying the set of negative examples in the training set to reflect network information. Prior work in PPI prediction has argued

that better selection of negative samples in the training set could improve the model, with Zhang et al. [164] exploring a random-walk distance on the PPI graph to distinguish between and low- and high-confidence negative examples. We explored the strategy of selecting only protein pairs with low GLIDE scores as negative examples, but found the performance to be poorer than the baseline. Drilling down, we found that this was due to a reduction in diversity of negative examples available for training, since using graph-theoretic measures to select negative examples restricts us to nodes occurring in the training PPI network (Figure 4.5, Section 4.4.9). In contrast, our incorporation of GLIDE scores in the objective allows us to handle a broader set of negative examples.

4.4.3 Cross-species improvement is not limited to hub nodes

Noting that Topsy-Turvy makes use of global PPI organization in the training phase but makes predictions solely using sequence data, we sought to characterize the kind of topological knowledge being learned by the trained model. Specifically, we investigated if the performance improvement of Topsy-Turvy over D-SCRIPT was limited to certain categories of proteins/nodes.

Since network-based methods work by learning network connectivity patterns, and some network structure is conserved across species, such methods tend to work well for proteins that already have many known interactions. Thus, it could be possible that the outperformance of Topsy-Turvy comes exclusively or primarily from, say, hub nodes whose interactions may be better conserved across species. To investigate this, we evaluated human-PPI trained Topsy-Turvy and D-SCRIPT on physical interactions in *D. melanogaster*, sourcing the latter from BioGRID (we found BioGRID's fly PPI annotations clearer than STRING's). Limiting ourselves to fly proteins that occur in the PPI network, we partitioned the fly evaluation set into four sub-groups by degree: each putative edge (p, q) was grouped as per $\mathcal{M}_{pq} = max(d(p), d(q))$, where $d(p)$ and $d(q)$ are the degrees of p and q in the fly PPI network, respectively. Thus, the sub-group corresponding to $\mathcal{M} \geq 21$ consists of putative interactions

where at least one of the proteins is a hub-like protein.

Even though baseline D-SCRIPT is not explicitly informed about network structure, it too demonstrated better performance as $\mathcal{M}$ increased. This may be due to the information encoded in the frequency with which each protein appears in the positive examples D-SCRIPT is trained on. Because of that, along with stronger conservation of PPIs involving hub nodes [165], [166], some network aspects can be implicitly learned by a purely sequence-based approach like D-SCRIPT. This also illustrates one of the core points of this paper— the connection between bottom-up and top-down views of protein interaction.

We also observed that Topsy-Turvy improved upon D-SCRIPT in each sub-group, indicating that the outperformance is not only coming from high-degree nodes. While Topsy-Turvy also achieves its highest performance on the $\mathcal{M} \geq 21$ sub-group, its improvement over D-SCRIPT is not limited to the highest-degree hub nodes. In fact, the relative AUPR improvement of Topsy-Turvy over D-SCRIPT is 2.22-fold when $\mathcal{M}$ is in the 2–20 range, compared to a 1.31-fold improvement for hub nodes ($\mathcal{M} \geq 21$) (Table 4.4). Topsy-Turvy thus not only improves predictive performance for high-degree nodes, but the GLIDE loss term additionally informs the model about global structure, leading to improvement for more sparsely connected nodes.

Table 4.4: **Cross-species performance of D-SCRIPT and Topsy-Turvy, subdivided by node degree in target species.** Both methods were trained on human PPI data and tested on fly (BioGRID). The analysis is limited to protein pairs where both proteins occur in the fly PPI graph. In addition to overall AUPR, we also group each protein pair by the maximum of the degrees of its nodes in the fly PPI network. Both methods improve as maximum degree increases, and Topsy-Turvy consistently outperforms D-SCRIPT across all subsets — especially so for putative interactions between low-degree nodes.

Model	**Overall AUPR**	**AUPR by Maximum Degree**			
		$2-5$	$6-10$	$11-20$	≥ 21
D-SCRIPT	0.356	0.030	0.067	0.118	0.475
Topsy-Turvy	**0.538**	**0.073**	**0.168**	**0.237**	**0.622**

4.4.4 Performance is unlikely to be driven by ascertainment bias

In the setting where bottom-up sequence methods are compared to top-down network-based methods (or synthesis approaches like Topsy-Turvy), issues of ascertainment bias [144] in the available ground truth network data become particularly acute. The issue is a simple one: existing PPI network data in all organisms (with the possible exception of recently-described HuRI [167]) is biased towards pairs of proteins a biologist decided to experimentally test for interaction, and biologists are more likely to include proteins already known to be of interest, or nodes that are already adjacent to other previously studied nodes in the network. The result is that nearly all ground-truth existing networks will over-estimate the performance of methods that incorporate network information, and under-estimate the performance of methods that utilize only sequence information, since missing edges are more likely to be falsely scored as negatives for the sequence based methods. When comparing network methods against network methods, or sequence methods against sequence methods, the respective alternative is likely to be similarly biased, making it less of a concern. However, when comparing methods across both types of information, addressing the bias becomes more important.

Our results in Section 4.4.3 begin to address the issue of ascertainment bias. Although the BioGRID *D. melanogaster* network is not fully unbiased, if the improvement of Topsy-Turvy over D-SCRIPT were coming only from this bias, we would expect to see disproportionate improvement in the dense core of the network, where interactions are most likely to be experimentally tested. Instead, we see improvement across the network, which suggests that Topsy-Turvy's cross-species performance gains come from successfully learning global network organization properties rather than suffering from ascertainment bias. We discuss the issue of this bias and how it might be addressed by future methods further in Section 4.5.

4.4.5 Comparison with AlphaFold-Multimer

We next investigated if recent advances in protein structure determination [133] that have enabled extremely high-quality protein complex structure prediction (in particular, AlphaFold-Multimer), could be leveraged for PPI prediction. While these methods were not designed to directly address *if* two proteins interact — they only predict the putative complex structure *assuming* an interaction — we investigated if AlphaFold-Multimer could nonetheless be adapted for our PPI prediction setting. From AlphaFold-Multimer results, we obtained their reported ipTM (interface predicted template modeling) score, a value between 0 and 1, that was shown in the original study to be correlated with the quality of the docked complex (DockQ score). For each candidate protein pair, we compute its mean ipTM score over the five AlphaFold-Multimer models. In our evaluations, we used this score as a predictor of protein interaction and assessed AlphaFold-Multimer on PPIs from the STRING *D. melanogaster* testing set used in Section 4.4.2.

We find that AlphaFold-Multimer is several orders of magnitude slower than Topsy-Turvy, requiring an average of 6 hours per pair (AlphaFold-reported time, min = 2.87hr, mean = 5.89hr, max = 12.97hr) compared to 0.02 seconds per pair for Topsy-Turvy (hardware described in Section 4.3.10). Of the total AlphaFold-Multimer runtime, an average of 3.22 hours were spent on feature generation (min = 1.62hr, max = 8.34hr) and 2.66 hours were GPU time spent on model computation (min = 1.16hr, max = 4.64hr). We note that feature generation time cannot necessarily be amortized over input pairs, since an important part of adapting AlphaFold to protein complexes is the proper alignment of paired multiple sequence alignments (MSAs) for each candidate protein pair. Thus, AlphaFold-Multimer is infeasible for genome-scale *de novo* PPI prediction for organisms with limited experimental data.

We compared AlphaFold-Multimer PPI predictions with those of Topsy-Turvy in a small-scale study, constrained by the computational requirements of AlphaFold-Multimer. We selected 18 candidate pairs that span the range of Topsy-Turvy scores as well as ground-

truth labels: six protein-pairs each with high (≥ 0.8), medium ($0.25 \leq \hat{y} < 0.8$), or low (≤ 0.25) Topsy-Turvy prediction scores, with three truly interacting and three non-interacting pairs in each subset. We note that distribution of Topsy-Turvy scores on these pairs is not representative of their full-sample distribution; for example, we expressly included examples where Topsy-Turvy was very confident but wrong, even though such instances comprise a small part of the broader distribution (89.8% of Topsy-Turvy scores are < 0.05). We found general agreement between AlphaFold-Multimer and Topsy-Turvy's predictions (Pearson's $\rho = 0.310$), though there were examples where each method correctly predicted an interaction that the other missed. We show full results in Table 4.5. Compared to Topsy-Turvy, AlphaFold-Multimer's scores seem calibrated for fewer false positives and more false negatives. In particular, AlphaFold-Multimer only scored two pairs with probability ≥ 0.8 both of which were true positives and also had high Topsy-Turvy scores; all other pairs were scored under 0.45. On three Topsy-Turvy false positives where it was highly confident but incorrect, AlphaFold-Multimer ipTM scores were low (mean = 0.3676). Conversely, AlphaFold-Multimer had substantial false negatives, missing three true interactions pairs that Topsy-Turvy correctly identified with medium or high probability. For pairs that Topsy-Turvy scored low, AlphaFold-Multimer agreed with it, with low ipTM scores (mean = 0.365).

These results suggest that Topsy-Turvy and AlphaFold-Multimer can each fill a valuable niche for predicting PPIs. Due to its low FPR, AlphaFold can be used to verify shortlisted interactions and accurately determine their complex structure. However, due to its run time constraints, it is infeasible to use for genome-scale predictions, a domain for which Topsy-Turvy would be more suitable. Additionally, the ipTM score is more a measure of complex stability than a predicted probability of interaction. Future work could seek to adapt the AlphaFold-Multimer architecture to explicitly address the PPI *prediction* task. For example, the calibration of interaction scores could be improved using insights gained from complete cross-docking approaches [168]. Recently, [169] have described physics-based energy, interface matching and protein sociability as useful metrics for identifying the likely

partners from an all-vs.-all docking study.

4.4.6 Integrative methods are applicable even in species with some available PPI data

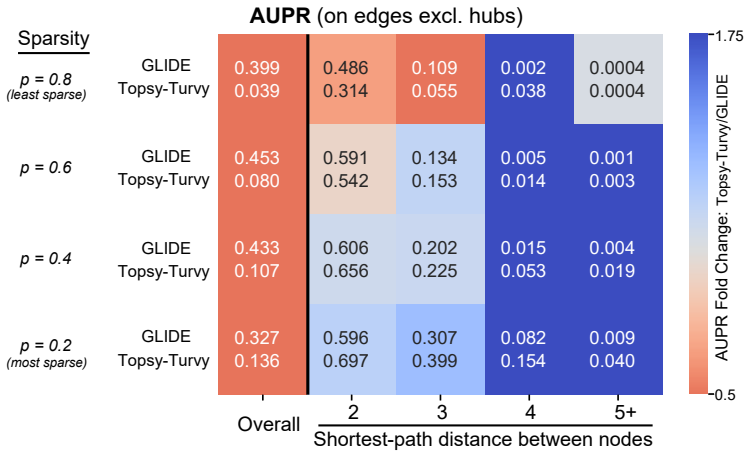


Figure 4.3: **Comparing Topsy-Turvy and GLIDE in situations when both can be used.** GLIDE was trained on a subset of the fly PPI network (e.g. training on 80% of PPIs when $p = 0.8$); Topsy-Turvy was trained on human PPI data and had no access to fly data for training. Both methods were evaluated on held-out positives as well as a randomly sampled set of negative examples, where pairs containing proteins with degree ≥ 21 on the subset networks were removed from the held-out examples during testing; the analysis is limited to proteins in the fly PPI network. In addition to reporting overall AUPR, we also group each protein-pair in the evaluation set by their shortest-path distance in the training network.

We have shown that human-trained Topsy-Turvy improves on human-trained D-SCRIPT when predicting PPIs in an organism using only sequence information (Sections 4.4.2-4.4.4). In non-model organisms, there might not be any experimentally tested physical interaction data— this is the situation for which D-SCRIPT was designed, and for which we have thus far tested Topsy-Turvy. However, we are also interested in applying Topsy-Turvy to predict PPIs in the case where some sparse network does exist in the species of interest. Specifically, we ask the following question: if some network edges exist in the target species of interest, should one use a purely network-based method, or a synthesis method like Topsy-Turvy when predicting new PPIs? Sequence-based synthesis methods are necessary to attach previously unseen

proteins to an existing network, but either method could be used to predict new interactions between proteins already in the network. Here, we show that a hybrid of Topsy-Turvy and GLIDE (TT-Hybrid, Section 4.3.7) improves upon either method alone in the case where some sparse network is available.

We consider situations where both proteins in the pair of interest occur in the PPI network, so that a network-only prediction can be made. Here, we evaluate GLIDE, Topsy-Turvy, and TT-Hybrid on the *D. melanogaster* BioGRID network, which has been partitioned to measure the performance on networks of varying sparsity characterized by a parameter $p \in \{0.8, 0.6, 0.4, 0.2\}$. More specifically, p describes the fraction of total edges in G used to construct a subset network $G_p = (V, E_p)$. Full details on the construction of G_p are in Section B.1. Characteristics of the sparse network data sets are described in Section B.2. The sparsified network G_p is then used to compute GLIDE scores.

To construct the test set at different p-values, we (a) selected the set of positive edges S_p^+ as all edges in G left out during the construction of G_p, i.e., $S_p^+ = E \setminus E_p$, and (b) randomly sampled negative examples from the set $(V \times V) \setminus E$ to obtain S_p^-. The test set $S_p = S_p^+ \cup S_p^-$ was used to evaluate the performance of D-SCRIPT and Topsy-Turvy (trained on human), and GLIDE (trained on G_p) (AUPRs in Section B.3). We also broke down the analysis into subsets of the evaluation set, based on shortest-path distance d in G_p connecting the two proteins. Our intuition here was to check the relative performance of these methods on closely- vs. distantly-connected proteins. Detailed descriptions of the training network G_p and the test data sets S_p are provided in Tables B.1 and B.2.

Upon initial investigation, we found that while GLIDE outperformed Topsy-Turvy overall, their relative performance on a protein pair depended on the shortest-path distance between the proteins (Table B.3). Since GLIDE performance is primarily driven by hubs, to more clearly investigate relative performance we then performed the same set of evaluations after removing any edges incident upon hubs (i.e., (u, v) where $(degree(u) \geq 21) \vee (degree(v) \geq 21)$). We then observed that Topsy-Turvy was stronger on nearly every subset of data (Figure 4.3).

However, GLIDE still performed better than Topsy-Turvy overall.

These results indicate that while GLIDE is able to separate PPIs by their network distance (which strongly correlates with whether or not there will be a reported interaction), once separated by network distance, Topsy-Turvy is able to finely organize similarly-distant proteins using the information gleaned from sequence and structure. Thus, we introduced TT-Hybrid, which uses GLIDE and Topsy-Turvy to partition PPIs both coarsely and finely. We show in Table 4.6 that TT-Hybrid improves upon either component method alone, achieving the highest overall AUPR on the fly network at all levels of sparsity (with hub nodes included).

4.4.7 TT3D outperforms previous methods

We evaluate TT3D in the same cross-species setting where D-SCRIPT and Topsy-Turvy were originally tested. Following [11], TT3D was trained and validated on known human PPI from the STRING database [163], filtered for experimentally determined physical binding interactions.

Then, the best model trained on human PPIs was tested on known interactions from other model organisms such as mouse (*Mus musculus*), fly (*Drosophila melanogaster*), roundworm (*Caenorhabditis elegans*), *Escherichia coli* and brewer's yeast (*Saccharomyces cerevisiae*), also from STRING. Sequences were clustered with human sequences at 40% similarity using CD-HIT [51] and those with high similarity to proteins in the training set were removed. We measure model performance using the area under the precision-recall curve (AUPR). The test sets were constructed to have a 1:10 ratio of positives to negatives, so a random method would have an AUPR of $1/11 \approxeq 0.09$.

We compare TT3D to D-SCRIPT and Topsy-Turvy, neither of which incorporate structural information, and find that augmenting the Topsy-Turvy model with the encoded 3Di Foldseek sequence improves its PPI predictions. We also test against simple sequence and structure homology based approaches. In Figure 4.4, we show precision-recall curves for each of the three deep learning methods on the five benchmark test sets. TT3D performs significantly better

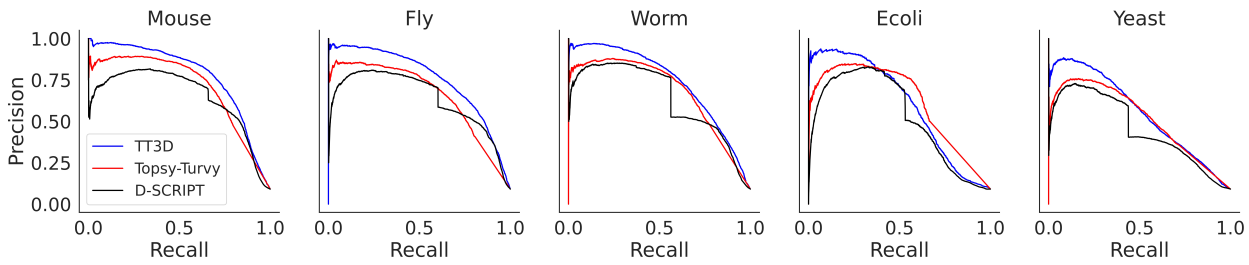


Figure 4.4: **Precision-Recall curves for TT3D, Topsy-Turvy, and D-SCRIPT.** Our experiments in organisms: Mouse, Fly, Roundworm, E. coli and Brewer's Yeast show TT3D substantially outperforming the other methods while predicting unknown PPI interactions.

than the other methods for all organisms that we tested on. In addition to overall performance, early precision (i.e., precision at low recall) is important, because often only a small number of highly predicted interactions are selected for downstream experimental prediction. We find that the precision values at low recall are closer to 1 for TT3D, which indicates that its top predictions are much more accurate than both D-SCRIPT and Topsy-Turvy.

4.4.8 Comparing TT3D's performance with simple sequence and structure homology transfer approaches

Sequence or structure-based homology approaches can also be used to transfer PPI annotations across species. We benchmarked TT3D against two such approaches, one based on Ensembl-provided sequence homology [170], and the other based on structural homology inferred using a pipeline of AlphaFoldDB, Foldseek and MMseqs2. We note two major challenges with such annotation transfer approaches. First, due to the bias in how candidate PPIs were chosen for assays, just knowing that a pair of target proteins have human homologs turns out to be a surprisingly good predictor of their interaction, achieving precision (recall) of 0.2065 (0.658) and 0.2313 (0.4442) in fly and yeast, respectively. Second, such an approach does not provide a probability of an interaction, so neither an average precision nor precision-recall curve can be computed. Nonetheless, we compared TT3D to these approaches by generating the exhaustive set of fly (or yeast) PPI candidates by considering all possible transfers of

human PPIs and scored these against ground-truth PPIs. TT3D outperformed both sequence and structure-based annotation transfer, achieving about 5x and 17x greater precision in fly than sequence and structure-based approaches, respectively (Table 4.7).

To determine sequence homologs, We used Ensembl-provided homology mappings. We originally considered, but then rejected the use of only one-to-one mappings, i.e., where a single fly protein is mapped to just one human protein, which yielded only 68 mappings between human and fly PPIs. Our subsequent analyses therefore were in the many-to-many homology mapping regime. To determine structure homologs, we used structures from AlphaFold DB, obtained the 3Di sequence representations from Foldseek, and then applied MMSeqs2 (with an e-value threshold of 10^{-10}) to map structural homologs between fly (or yeast) and human.

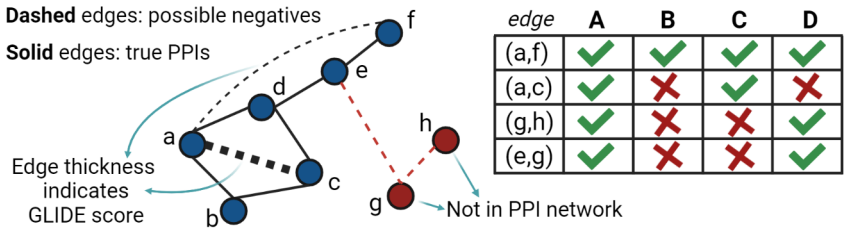
We took the entire set of human PPIs from our training set and mapped them to all potential fly (or yeast) homolog pairs, reasoning that this would be the exhaustive set of potential fly PPIs that any annotation transfer method could potentially consider as true. We note that this mapping resulted in a massive expansion of potential fly PPIs. The $\sim$ 38,000 human training-set PPIs were mapped to $\sim$180,000 fly PPIs (Ensembl-based homology mapping) and $\sim$3.1 million fly PPIs (structure-based homology mapping). We considered the union of these with fly pairs with the true positive fly pairs ($\sim$27,000 from STRING v11; physical binding interactions only). We created analogous networks for yeast, observing that the set of homologs between human and yeast is smaller. Because of distinct sequence and structure-based homology mapping schemes, these evaluations were essentially on two separate datasets, which we denote as "Networks from Sequence Homology" and "Networks from Structure Homology", respectively.

We applied TT3D on these datasets, computing its precision and recall curves. For homology-based annotation transfer, precision was computed as standard:

$$\text{Intersection}(\textit{True\_Positives, Predicted\_Positives})/\textit{Predicted\_Positives}. \tag{4.7}$$

Here, *True_ Positives* are the ground-truth fly PPIs while *Predicted_ Positives* is the set of PPIs mapped from human. Recall was similarly computed as per its standard definition. We note that annotation transfer provides only one set of precision and recall scores, while TT3D's score (in the range 0–1) can be thresholded to provide a range of precision-at-desired-recall scores and the corresponding Areas Under Precision-Recall Curve.

4.4.9 Considering network structure for negative sample selection has marginal impact



	Negative Sampling Method	AUPR	AUROC
A	All nodes; random pairs	0.605	0.890
B	PPI nodes only; weak GLIDE scores	0.380	0.840
C	PPI nodes only; random pairs	0.477	0.832
D	All nodes; exclude strong GLIDE scores	**0.629**	**0.913**

Figure 4.5: **Using network information to guide selection of negative training examples.** The common practice in PPI prediction literature is to consider random pairs of proteins as negative examples (row **A**). Restricting negative examples to just low-scoring GLIDE node pairs (row **B**) results in substantially weaker performance, likely due to reduced diversity of negative examples. Ablation studies (rows **C** and **D**) support the hypothesis. AUPR and AUROC were computed using a D-SCRIPT model trained on human protein interactions, with negative edges generated by each respective method, and evaluated on the *D. melanogaster* STRING network.

We assessed if using GLIDE scores to select negative training examples results in higher quality training data and improved model performance. We first applied GLIDE on the PPI network formed by positive examples in the training set, calculating a GLIDE score for all pairwise combinations of nodes (i.e. proteins) in this network. The negative examples were then chosen from these as the k lowest-scoring protein pairs; as in the previous experiments,

we chose k to ensure a 10:1 negative-to-positive ratio. On this modified training set, we trained a vanilla D-SCRIPT model (i.e., no network loss in the objective) on human PPI data and evaluated it on *D. melanogaster* interactions.

Surprisingly, the baseline version of D-SCRIPT (trained on negative examples chosen completely at random) had substantially stronger performance than the version of D-SCRIPT trained on a network-informed training set (rows A vs. B in Figure 4.5). We wondered if the latter's lower performance was due to a reduced diversity of negative examples in the latter: using GLIDE to select negative examples limits us to protein pairs where both proteins exist in the PPI network. Since experimental PPI data is limited, the negative examples are restricted to only a subset of the human proteome. In contrast, the baseline version of D-SCRIPT includes negative examples where one or both the proteins might not occur in the positive examples. To test our hypothesis, we trained and evaluated two other variants: i) choose negative examples completely at random, but limit the proteins to those occurring in the PPI network, and ii) in the baseline training set, remove all negative examples (p, q) where p and q both occur in the PPI network and replace them with an equal number of examples (p', q') where p' and q' also occur in the PPI network but (p', q') has a low GLIDE score. As we hypothesized, the first variant (row C in Figure 4.5) performed worse than the baseline D-SCRIPT (row A). The second variant (row D) performed marginally better than the baseline, suggesting that the incorporation of network information in the training set construction does help somewhat. However, this improvement was marginal and unclear (for instance, row C has a lower AUROC than row B), so we chose not to incorporate it into Topsy-Turvy.

4.5 Chapter Perspectives

We have presented Topsy-Turvy, a new method that integrates top-down global view of PPI organization into a bottom-up sequence-based PPI prediction model. The neural network

design of Topsy-Turvy builds upon the architecture of D-SCRIPT and, like the latter, includes a bottleneck layer designed to model the inter-protein contact map, thus offering interpretability and insight into the mechanism of interaction. We show that Topsy-Turvy is highly accurate in a cross-species context, and applicable to species with few or no known protein interactions. For cases where PPI data is available in the target species, we present TT-Hybrid, that can leverage this additional information for more accurate predictions.

Topsy-Turvy thus improves upon the state-of-the-art in PPI prediction broadly— both in species without available PPI data and in those with PPI data. For the former, it is able to transfer knowledge of network structure from other species, leading to more accurate *de novo* predictions. For the latter, it improves prediction coverage as well as accuracy. For instance, even in well-studied species like human, mouse, and fly, there remain many proteins with no characterized PPIs (24.9%, 44.9% and 19.8% of proteins in the three species, respectively [171], [172]). Topsy-Turvy can be used to attach these hitherto uncharacterized proteins to existing PPI networks. Since GLIDE and other network methods are limited to predicting links between proteins that both already exist in the network, they cannot be used for putative interactions involving such proteins. When both proteins do exist in the PPI network, the hybrid approach TT-Hybrid that combines GLIDE with Topsy-Turvy performs better than either approach alone, with the former achieving a coarsely accurate network-theoretic organization and latter fine-tuning it locally. Here, we hypothesize that GLIDE confers species-specific network information unable to be transferred by Topsy-Turvy due to network rewiring.

The TT-Hybrid results also give some hint as to what Topsy-Turvy might be learning from including a network loss term in the *training* stage. As shown in Figure 4.3, the GLIDE network score helps segregate proteins into buckets that give a macro range of potential probabilities that an edge exists, while the bottom-up sequence approach does best at ranking the specific pairs within each bucket. This is not the first time we have seen network based information assist in making sequence-level information more accurate; the Isorank network

alignment algorithm [173] also receives a gain in performance in discovering orthologs by a global top-down network similarity score that augments the bottom-up pairwise sequence score.

In this regard, Topsy-Turvy presents an approach to a often-faced challenge in systems biology: how to resolve the dichotomy between a bottom-up and top-down view of the same biological phenomenon? Considered at the molecular level, protein interaction is a purely physicochemical process. However, these proteins primarily function through their interactions. With proteins performing most of the functions in the cell, evolution constrains the space of possible protein folds, resulting in emergent properties at the network level. The approach embodied by Topsy-Turvy and TT-Hybrid could be more generally applied to situations where network-theoretic and molecular views need to be integrated. To make a social interaction analogy, D-SCRIPT and other sequence-based bottom-up methods are learning features that make two people likely to be compatible as friends, but not global organization of the friend network that would indicate if those two people share enough mutual friends to be likely to have had the opportunity to meet at the same event.

While we took steps to rule out the effect of ascertainment bias, this remains an important question in both the training and evaluation of link prediction methods. In this work, we sourced PPIs from the STRING database where data from a variety of assays has been conglomerated. An unbiased, all-vs-all screen as exemplified by the Human Reference Interactome (HuRI) database [167] offers the promise of addressing ascertainment bias in the specific case of yeast two-hybrid (Y2H) screens. However, to test Topsy-Turvy in our transfer-learning context, we would also need similar unbiased Y2H screens in a different species.

By approaching integration of orthogonal information sources as a multi-objective learning problem, Topsy-Turvy lays the groundwork for incorporation of additional data modalities. For instance, while the GLIDE score incorporates both global and local scores, it would be possible to directly supervise Topsy-Turvy with global and local loss terms, each with a

respective hyper-parameter to finely control their effects. Loss terms that quantify protein functional similarity [174] or interface similarity [175], [176] could be added to the framework to further inform predictions. Topsy-Turvy demonstrates that a general, scalable framework that allows us to transfer both low-level (sequence-to-structure) and high-level (network topology) insights across species can enable researchers to fill in the missing links in our knowledge of biological function.

We additionally introduce Topsy-Turvy 3D (TT3D), which builds off of prior work in sequence-based PPI prediction [12] to incorporate structure by jointly modeling both amino acid sequence and 3Di sequence. We demonstrate that TT3D is able to take advantage of the compact representation of protein structure to improve the accuracy of PPI prediction in a cross-species context. In an era where high-quality predictions of protein structure are readily available for many proteins, we expect that TT3D can be easily substituted into pipelines which use lightweight sequence-only deep learning prediction methods to make high-quality predictions, while remaining fast enough to be applied at genome scale.

Table 4.5: We report the predicted probability of interaction from Topsy-Turvy and AlphaFold-Multimer (Mean ipTM over 5 models) for 18 candidate fly protein pairs. We also report the full run time of AlphaFold-Multimer in seconds, as well as the times for HMM search and feature generation (Feature Time) and total prediction time over five models (GPU Time). Each Topsy-Turvy prediction takes approximately 0.02 seconds.

ID1	ID2	Interaction Label	Topsy-Turvy Score	Mean ipTM	Full Time (s)	Feature Time (s)	GPU Time (s)
7227.FBpp0078386	7227.FBpp0112233	0	0.8639	0.2453	24714.69	17229.83	7484.85
7227.FBpp0071140	7227.FBpp0297773	0	0.8824	0.2152	22462.42	6670.02	15792.40
7227.FBpp0072955	7227.FBpp0080316	0	0.9715	0.3169	19735.48	8131.67	11603.80
7227.FBpp0088242	7227.FBpp0088439	1	0.8319	0.3660	31796.50	24201.23	7595.26
7227.FBpp0085619	7227.FBpp0086223	1	0.8383	0.8059	11169.22	6975.87	4193.35
7227.FBpp0113041	7227.FBpp0290012	1	0.8270	0.8503	13510.71	6942.38	6568.33
7227.FBpp0080022	7227.FBpp0303413	0	0.0004	0.3939	18286.96	6986.66	11300.30
7227.FBpp0078625	7227.FBpp0086780	0	0.0001	0.3786	21367.15	10417.71	10949.43
7227.FBpp0071149	7227.FBpp0074705	0	0.0003	0.1734	22037.37	6207.01	15830.36
7227.FBpp0071136	7227.FBpp0088946	1	0.0004	0.2895	20386.57	10035.56	10351.00
7227.FBpp0079704	7227.FBpp0304061	1	0.0102	0.2173	19813.02	10494.35	9318.66
7227.FBpp0085164	7227.FBpp0087553	1	0.0026	0.3683	20318.73	7338.50	12980.23
7227.FBpp0072055	7227.FBpp0073836	0	0.3921	0.3157	46704.21	30016.38	16687.82
7227.FBpp0083802	7227.FBpp0087849	0	0.4864	0.2269	19511.20	13965.89	5545.31
7227.FBpp0073028	7227.FBpp0086215	0	0.7401	0.2924	13408.14	7660.80	5747.34
7227.FBpp0071046	7227.FBpp0079780	1	0.6129	0.3442	33131.66	22314.30	10817.35
7227.FBpp0082370	7227.FBpp0100175	1	0.2989	0.3197	12807.16	7447.19	5359.96
7227.FBpp0076184	7227.FBpp0079616	1	0.4962	0.4495	10344.67	5836.69	4507.97

Table 4.6: **TT-Hybrid improves upon both of its constituent components on in-species prediction.** We generated partitions of the fly network of varying sparsity, using the sparsified networks as training for GLIDE. Sparsity p corresponds to the proportion of edges retained in the training network ($p = 0.8$ is the least sparse). Topsy-Turvy was trained on human PPIs. TT-Hybrid combines the predictions from both GLIDE and Topsy-Turvy. Here we report the AUPR of each method on the held out edges removed from each network subset. We also show the AUPR of the random control; due to varying class imbalances, AUPR scores increase slightly with increasing sparsity.

Sparsity	**GLIDE**	**Topsy-Turvy**	**TT-Hybrid**	**Random**
$p = 0.8$	0.380	0.038	**0.387**	0.004
$p = 0.6$	0.437	0.079	**0.451**	0.009
$p = 0.4$	0.412	0.105	**0.423**	0.014
$p = 0.2$	0.318	0.133	**0.354**	0.019

Table 4.7: **Comparing TT3D to annotation transfer with sequence or structural homology.**

Experiment	**Species**	**Predictions By**	**Recall**	**Precision**	**AUPR**
-	Fly	TT3D	-	-	0.6263
-	Yeast	TT3D	-	-	0.4755
Homology Only	Fly	Sequence	0.658	0.2065	-
Homology Only	Fly	Structure	0.7864	0.2967	-
Homology Only	Yeast	Sequence	0.4442	0.2312	-
Homology Only	Yeast	Structure	0.8438	0.1386	-
Homology+Interaction in Human	Fly	Sequence	0.384	0.99	-
Homology+Interaction in Human	Fly	Structure	0.6396	0.7894	-
Homology+Interaction in Human	Yeast	Sequence	0.1922	0.9688	-
Homology+Interaction in Human	Yeast	Structure	0.5126	0.4879	-
Networks from Sequence Homology	Fly	TT3D	0.3858	0.2934	0.2812
Networks from Sequence Homology	Fly	Sequence	0.3858	0.0598	-
Networks from Sequence Homology	Yeast	TT3D	0.1917	0.6913	0.7409
Networks from Sequence Homology	Yeast	Sequence	0.1917	0.4025	-
Networks from Structure Homology	Fly	TT3D	0.6452	0.0604	0.0759
Networks from Structure Homology	Fly	Structure	0.6452	0.0056	-
Networks from Structure Homology	Yeast	TT3D	0.5397	0.0856	0.0583
Networks from Structure Homology	Yeast	Structure	0.5397	0.0131	-

Chapter 5

Protein-Small Molecule Interactions: Learning Shared Representations

5.1 Chapter Overview

In time and money, one of the most expensive steps of the drug discovery pipeline is the experimental screening of small molecules to determine binding to a protein target of interest. Therefore, accurate high-throughput computational prediction of drug-target interactions would unlock significant value, guiding and prioritizing promising candidates for experimental screening. In this chapter, we introduce ConPLex, a machine learning method for predicting drug-target binding which achieves state-of-the-art accuracy on many types of targets by using a pretrained protein language model. The approach co-locates the proteins and potential drug molecules in a shared feature space while learning to contrast true drugs from similar nonbinding "decoy" molecules. ConPLex is extremely fast, which allows it to rapidly shortlist candidates for deeper investigation.

5.2 Introduction

In the drug discovery pipeline, a key rate-limiting step is the experimental screening of potential drug molecules against a protein target of interest. Thus, fast and accurate computational prediction of drug-target interactions (DTIs) could be extremely valuable, accelerating the drug discovery process. One important class of computational DTI methods, molecular docking, uses 3D structural representations of both the drug and target. While the recent availability of high-throughput accurate 3D protein structure prediction models [133], [146], [148] means that these methods can be employed starting only from a protein's amino acid sequence, the computational expense of docking [177] and other structure-based approaches (e.g., rational design [178], active site modeling [179], template modeling [131], [180]) unfortunately remains prohibitive for large-scale DTI screening. An alternative class of DTI prediction methods use 3D structure only implicitly, making rapid DTI predictions when the inputs consist only of a molecular description of the drug (such as the SMILES string [181]) and the amino acid sequence of the protein target. This class of *sequence-based* DTI approaches enables scalable DTI prediction, but there have been barriers to matching the levels of accuracy obtained by structure-based approaches.

In this paper we introduce ConPLex, a new rapid purely sequence-based DTI prediction method that leverages rich featurizations from pre-trained protein language models (PLMs), and show it can produce state of the art performance on the DTI prediction task at scale. The advance provided by ConPLex comes from two main ideas that together overcome some of the limitations of previous approaches: informative PLM-based representations and contrastive learning. While many methods have been proposed for the sequence-based setting of the DTI problem [182] (e.g., using secure multi-party computation [183], convolutional neural networks [184] or transformers [185]), their protein and drug representations are constructed solely from DTI ground truth data. The high level of diversity among the DTI inputs, combined with the limited availability of DTI training data, limit the accuracy of these methods and their

generalizability beyond their training domain. Furthermore, the methods that do generalize often do so by sacrificing fine-grained specificity, i.e., are unable to distinguish true-positive binding compounds from false positives with similar physico-chemical properties ("decoys").

In contrast, the "PLex" (Pre-trained Lexographic) part of ConPLex helps alleviate the problem of limited DTI training data. As we showed in our preliminary work [186], one way to get around the limited size of DTI data sets that has hampered the quality of the representations learnt by previous methods is to transfer learned proteins representations from pre-trained protein language models to the DTI prediction task. PLMs learn the distributional characteristics of amino acid sequences over millions of proteins in an unsupervised fashion, generating sequence-based representations that encode deep structural insights. A design paradigm in machine learning is that an informative featurization of the input can enhance the power of even simple models. For drug-target interaction, where task-specific data is limited, using PLM-generated representations as the input features allows us to borrow strength from the much larger corpus of single protein sequences [186]. Starting with the PLM models, our second insight directly addresses the fine-grained specificity problem in our architecture by using the "Con" (Contrastive learning) part: a novel, protein-anchored contrastive co-embedding that co-locates the proteins and the drugs into a shared latent-space. We show that this co-embedding enforces separation between true interacting partners and decoys to achieve both broad generalization and high specificity.

Putting these two ideas together gives us ConPLex, a novel representation learning approach that enables both broad generalization and high specificity. We show that ConPLex enables more accurate prediction of DTIs than competing methods while avoiding many of the pitfalls suffered by currently available approaches. Thus, our work constitutes a concrete demonstration of the power of a well-designed transfer learning approach that adapts foundation models for a specific task [187], [188]. In particular, we found that the performance of existing sequence-based DTI prediction methods could be sensitive to variation in drug-vs-protein coverage in the data set, whereas ConPLex performs well in multiple

coverage regimes. Indeed, ConPLex performs especially well relative to other methods in the zero-shot prediction setting where no information is available about a given protein or drug at training time. Experimental validation of ConPLex yielded a 63% hit rate (12/19), including four hits with sub-nanomolar binding affinity, demonstrating the value of ConPLex as an accurate, highly-scalable, in silico screening tool.

ConPLex can also be adapted beyond the binary cases to make predictions about binding affinity. Furthermore, the shared representation also offers advantages beyond prediction accuracy. The co-embedding of both proteins and drugs in the same space offers intepretability, and we show that distances in this space meaningfully reflect protein domain structure and binding function: we leverage ConPLex representations to functionally characterize cell-surface proteins from the Surfaceome database [189], a set of 2,886 proteins localized to the external plasma membrane that participate in signaling and are likely able to be easily targeted by ligands.

ConPLex is extremely fast: as a proof-of-concept, we make predictions for the human proteome against all drugs in ChEMBL [190] ($\approxeq 2 \times 10^{10}$ pairs) in just under 24 hours using a single NVIDIA A100 GPU. Thus, ConPLex has the potential to be applied for tasks which would require prohibitive amounts of computation for purely structure based approaches or less efficient sequence-based methods, such as genome-scale side-effect screens, identifying drug re-purposing candidates via massive compound libraries searches, or *in silico* deep mutational scans to predict variant effects on binding with currently approved or potential new therapeutics. We note that most DTI methods require significant computation on each drug-target pair (i.e., have quadratic time-complexity). Because ConPLex predictions rely only on the distance in the shared space, predictions can be made highly efficiently once embeddings (which have linear time-complexity) are computed.

Distinguishing between low- and high-coverage DTI prediction We benchmark performance of ConPLex and competing methods in two different regimes, which we term

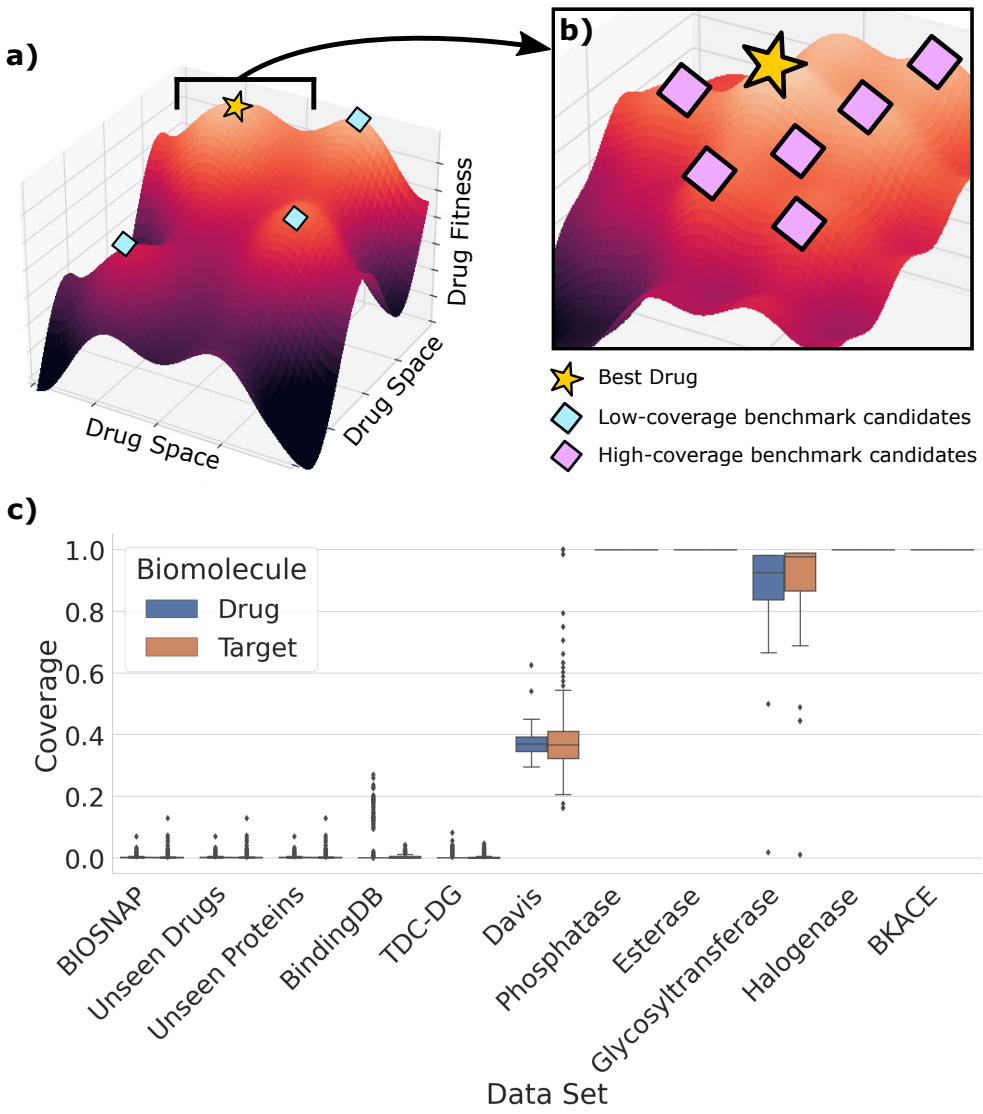


Figure 5.1: **Drug-target interaction benchmarks display highly variable levels of coverage.** Coverage is defined as the proportion of drugs or targets for which a data point (positive or negative) exists in that data set. High- vs. low-coverage benchmarks tend to reward different types of model performance. **(a)** In this cartoon of an example low coverage data set, drug candidates cover the full diversity of the space, and no two drugs are highly similar. A successful model can learn a coarse estimate of the fitness landscape, but must accurately model a large part of drug space to generalize to all candidates. **(b)** For high-coverage data sets, drugs tend to be targeted to a specific protein family. Thus, a successful model does not need to generalize nearly as widely, but must be able to capture more minor variation in drug fitness to achieve high specificity and differentiate between similar drugs. **(c)** In a review of existing popular DTI benchmark data sets, we find widely varying coverage, from data sets with nearly zero coverage (each drug/target is represented only a few times) to nearly full coverage (all drug-by-target pairs are known in the data).

low-coverage and high-coverage DTI prediction (Figure 5.1c). We show that ConPLex outperforms its competitors in both settings, but note that separating the two regimes helps clarify an often-seen issue in the field: methods whose performance varies substantially across different proposed DTI benchmarks. Several prior attempts have been made to standardize DTI benchmarking and develop a consistent framework for model evaluation [191], [192]. However, much of this work has overlooked a key aspect of benchmarking that we find to significantly affect model performance— differing per-biomolecule data coverage. We define coverage as the average proportion of drugs or targets for which a data point exists in that data set, whether that is a positive or negative interaction (Section 5.3.1). Depending on the per-biomolecule data coverage of the benchmark data set, we claim that these benchmarks are looking at very different problems. In particular, low-coverage data sets (Figure 5.1a) tend to measure the broad strokes of the DTI landscape, containing a highly diverse set of drugs and targets. Such data sets can present a modeling challenge due to the diverse nature of targets covered, but allow for a broad assessment of compatibility between classes of compounds and proteins. High-coverage data sets (Figure 5.1b) represent the opposite trade-off: they contain limited diversity in drug or target type, but report a dense set of potential pairwise interactions. Thus, they capture the fine-grained details of a specific sub-class of drug-target binding and enable distinguishing between similar biomolecules in a particular context.

The two coverage regimes correspond to different usage cases. The low-coverage regime is relevant when applying DTI models for large-scale scans to predict interactions for a potential target against a large compound library (e.g. for drug re-purposing as in Dönertaş et al. [193] and Morselli et al. [194]), or for scanning a candidate drug against an entire proteome to identify potential adverse and off-target effects (as in Huang et al. [195], [196]). Data at this scale is often low coverage, with only a small number of known interactions for each unique biomolecule. Thus, it is important that DTI models used for these tasks are broadly applicable and can accurately generalize to many different families of proteins and drugs. However, this generalization often comes at the cost of specificity, resulting in models that

are unable to distinguish between highly similar drugs or proteins.

The high-coverage regime is relevant when optimizing a particular interaction. Here, models can be trained to be highly specific to a protein family or class of drugs, so much so that a per-drug or per-target model is trained to capture the precise binding dynamics of that biomolecule [197]. While such models can be effective for lead optimization, they require high coverage on the biomolecule of interest to make accurate predictions; this may not always be available. Additionally, such models lack the capacity to generalize beyond the training domain and thus cannot be used for genome- or drug bank-scale prediction.

The PLM approach of ConPLex enables strong performance in both regimes. In the low coverage regime, the strength is coming mostly from the "PLex" part, where it can leverage the effective generalization of language models to achieve state-of-the-art performance. On high-coverage data sets, the "Con" part also becomes important, since it becomes feasible to train drug- or target-specific models with high accuracy, and such models often outperform more generic models. We find that while single-task models do perform well given available data, ConPLex is able to achieve extremely high specificity in low-diversity, high-coverage scenarios, while remaining broadly applicable to protein targets with limited data. Thus, ConPLex is applicable for both large-scale compound or target screens and fine-grained, highly specific binding prediction. We discuss the issue of matching the right model to the problem domain with respect to coverage further in Section 5.5.

5.3 Methods

5.3.1 Computing data set coverage

Let $1_{(i,j)}$ be the indicator variable meaning there exists an observation of drug i and target j. For a data set with m unique drugs and n unique targets, we can define the coverage for drug d as $C_d = \frac{1}{n}\sum_{j=0}^{n} 1_{(d,j)}$ and for a target t as $C_t = \frac{1}{m}\sum_{i=0}^{m} 1_{(i,t)}$. Then, for a given data set we can evaluate the median drug and target coverage. A data set with maximum coverage

would have a single data point for each drug-target pair, and thus a median coverage of 1 for both drugs and targets. Conversely, each drug and target would only be represented a single time in a minimum coverage data set, resulting in drug and target coverages of $\frac{1}{n}$ and $\frac{1}{m}$ respectively. We report the median drug and target coverage for each benchmark data set in Table 5.1. Since the DUD-E data set is separated out by targets, we instead report the median number of drugs against each target.

5.3.2 Benchmarks overview

Low coverage benchmarks We evaluate our framework on three broad-scale, low-coverage benchmark data sets. Two data sets, **DAVIS** [201] and **BindingDB** [200], consist of pairs of drugs and targets with experimentally determined dissociation constants (K_D). Following [185], we treat pairs with $K_D < 30$ as positive DTIs, while larger K_D values are negative. The third data set, ChG-Miner from **BIOSNAP** [199], consists of only positive DTIs. We create negative DTIs by randomly sampling an equal number of protein-drug pairs, making the assumption that a random pair is unlikely to be positively interacting. The DAVIS data set represents a few-shot learning setting: it contains only 2,086 training interactions, compared to 12,668 for BindingDB and 19,238 for BIOSNAP. The rest of the data preparation follows [185]. The data sets are split into 70% for training, 10% for validation, and the remaining 20% for testing. Training data are artificially sub-sampled to have an equal number of positive and negative interactions, while validation and test data is left at the ratio originally in the data set.

Zero-shot benchmarks. We evaluate our framework on two zero-shot prediction modifications of BIOSNAP. Following [185], the **Unseen proteins** set was created by selecting 20% of proteins from the full set, and selecting any interactions including these proteins for the test set. Thus, there are no proteins which appear in both the training and test set. The corresponding process was used to create the **Unseen drugs** data set. The training set was

then further split using 7/8 of the interactions for training and 1/8 of the interactions for testing. As above, data are sub-sampled so that training is balanced.

Continuous benchmarks. Continuous affinity prediction data come from the Therapeutics Data Commons DTI Domain Generalization benchmark (**TDC-DG**) [191]. The TDC-DG consists of 140,746 unique drugs and 477 unique targets derived from BindingDB [200] interactions that have patent information. Each interaction is labeled with an experimentally determined dissociation constant (IC_{50}). Interactions are temporally split, so that training pairs are from patents filed between 2013 and 2018, and test pairs are from between 2019 and 2021. 20% of the training pairs are randomly set aside as a validation set. We train 5 different models with the train/validation splits determined by the TDC benchmarking framework, and report the average Pearson correlation coefficient of predictions on the test set.

High coverage benchmarks. The Database of Useful Decoys: Enhanced (**DUD-E**) [207] consists of 102 protein targets and known binding partners (average 224 molecules per target). For each binding partner, there are 50 "decoys", or physio-chemically similar compounds that are known not to bind with the target. 57 of the targets are classified as either GPCRs, kinases, nuclear proteins, or proteases. We generate train-test splits by splitting targets within classes, so that there are representative members of each class in both the training and test set, but no target appears in both the training and test set (26 train, 31 test). These data are by definition high-coverage, since there are several true and decoy compounds available for each target. We provide the full target splits in Section C.1.

We also evaluate on several protein-family specific data sets from various different sources and compiled by Goldman et al. [197]. These include DTI data on β-ketoacid cleavage (**BKACE**) [206], **Esterase** [203], **Glycosyltransferases** [204], **Halogenase** [205], and **Phosphatase** [202] enzymes. These data are uniformly very high coverage, with a known data point for nearly every drug-target pair. Following [197], we performed a 10-fold cross

validation where the data were split into train-test sets by target, so that all drugs appear in both the training and test set, but no target does.

5.3.3 Overview of ConPLex

Target featurization

We generate protein target features using pre-trained protein language models (PLM): These models generate a protein embedding $E_{full} \in \mathbb{R}^{n \times d_t}$ for a protein of length n, which is then mean-pooled along the length of the protein resulting in a vector $E \in \mathbb{R}^{d_t}$. Specifically, we investigate the pre-trained models Prose [48], [53], ESM [208], and ProtBert [209], with default dimensions $d_t = 6165, 1280, 1024$ respectively (Section C.2.1). Elnaggar et al. recommend the use of ProtT5XLUniref50, but we found that it did not perform as well as ProtBert for the DTI prediction task. We emphasize that the language and projection models are used exclusively to generate input features– their weights are kept unchanged and are not updated during DTI training.

Drug featurization

We featurize the drug molecule by its Morgan fingerprint [210], an encoding of the SMILES string of the molecular graph as a fixed-dimension embedding $M \in \mathbb{R}^{d_m}$ (we chose $d_m = 2048$) by considering the local neighborhood around each atom. The utility of the Morgan fingerprint for small molecule representation has been demonstrated in [197], [211]. We additionally investigated the use of molecule embeddings from Mol2Vec [212] and MolR [213] and found they failed to perform as well as the Morgan fingerprint (Section C.2.2).

Transformation into a shared latent space and prediction

Given a target embedding $T \in \mathbb{R}^{d_t}$ and small molecule embedding $M \in \mathbb{R}^{d_m}$, we transform them separately into $T^*, M^* \in \mathbb{R}^h$ using a single fully-connected layer with a ReLU activation.

These layers are parameterized with weight matrices $W_t \in \mathbb{R}^{h \times d_t}, W_m \in \mathbb{R}^{h \times d_m}$ and bias vectors $b_t, b_m \in \mathbb{R}^h$.

$$T^* = ReLU(W_t T + b_t) \tag{5.1}$$

$$M^* = ReLU(W_m M + b_m) \tag{5.2}$$

Given the latent embeddings T^*, M^*, we compute the probability of a drug-target interaction $\hat{p}(T^*, M^*)$ as the cosine similarity between the embedding vectors, followed by a sigmoid activation. Thus, we compute the predicted probability as

$$\hat{p}(T^*, M^*) = \sigma(\frac{T^* \cdot M^*}{||T^*||_2 \cdot ||M^*||_2}) \tag{5.3}$$

When predicting compound binding affinity $\hat{y}(T^*, M^*)$, we substitute the sigmoid and cosine similarity (Equation 5.3) with a dot product followed by a ReLU activation, which gives a non-negative distance in the embedding space (Equation 5.4).

$$\hat{y}(T^*, M^*) = ReLU(T^* \cdot M^*) \tag{5.4}$$

Training

The model is trained both for broad and fine prediction, with the loss computed depending on the training data set. Broad-scale training data uses the binary cross-entropy loss (L_{BCE}) between the true labels y and the predicted interaction probabilities $\hat{p}$. When the model was trained to predict binding affinity, we substitute the binary cross-entropy loss for the mean squared error loss (L_{MSE}) is used during supervision.

Training on fine-scale data (DUD-E) was performed using contrastive learning. Contrastive learning uses triplets of training points rather than pairs, denoted the **anchor**, **positive**, and **negative**, and aims to minimize the distance between the anchor and positive examples while maximizing the distance between the anchor and the negative examples. In the DTI

setting, the natural choice for a triplet is the protein target as the anchor, the true drug as the positive and decoy as the negative example, respectively. We derive a training set of triplets in the following manner: for each known interacting drug-target pair (T, M^+), we randomly sample $k = 50$ non-interacting pairs (T, M^-) and generate the triplets (T, M^+, M^-), where M^- is drawn from the set of all decoys against T. We map these to latent space embeddings as described above. Since all the entities are now comparable to each other, we can compute the triplet margin-distance loss (L_{TRM}).

$$L_{TRM}(a, p, n) = \frac{1}{N} \sum_{i=1}^{N} max(D(a, p) - D(a, n) + m, 0) \tag{5.5}$$

where

$$D(u, v) = 1 - \hat{p}(u, v) \tag{5.6}$$

The margin m sets the maximum required delta between distances, above which the loss is zero.

Margin annealing

The margin m sets the maximum required delta between distances, above which the loss is zero. Initially, a large margin requires the decoy to be much further from the target than the drug to avoid a penalty, resulting in larger weight updates. As training progresses, lower margins relax this constraint, requiring only that the drug be closer than the decoy as $m \to 0$. Here, the margin is initialized at $M_{max} = 0.25$ according to a tanh decay with restarts decay schedule. Every $E_{max} = 10$ contrastive epochs, the margin is reset to the initial M_{max}, for a total of 50 epochs. At epoch i, the margin is set to

$$m(i) = M_{max}(1 - \tanh(\frac{2(i \bmod E_{max})}{E_{max}})) \tag{5.7}$$

Implementation

Model weights were initialized using the Xavier method from a normal distribution [214]. Weights were updated with error back-propagation using the AdamW optimizer [215] for a total of 50 epochs. For the binary classification task, the learning rate was initially set to 10^{-4} and adjusted according to a cosine annealing schedule with warm restarts [216] every 10 epochs. For the contrastive task, the learning rate was initially set to 10^{-5} and the same annealing schedule was followed. The margin for the contrastive loss was initially set to 0.25 and decreased to a minimum of 0 over 50 epochs according to a tanh decay schedule with restarts every 10 epochs. We used a latent dimension $d = 1024$ (results were robust to even with lower dimensions, and much higher dimensions may over-fit or be subject to topological restrictions) and a batch size of 32. The model was implemented in PyTorch version 1.11. Model training and inference was performed on a machine with a 112-core Intel Xeon Gold 6258R CPU and using a single NVIDIA A100 GPU. We compare training and inference run times in Section C.3.

5.3.4 Surfaceome analysis

We evaluate the functional use of ConPLex embeddings using data from the Surfaceome database [189], which contains 2,886 cell-surface proteins. We identified Pfam domains using HMMscan from HMMER3 [62] with default settings. We analyzed domains hit in > 10 proteins. For each domain, we trained a logistic regression classifier from sklearn with balanced class weights. We also evaluated domain coherence using spectral clustering with $k = 10$ clusters, and evaluated the adjusted mutual information (AMI) between true clusters (protein has/doesn't have domain) and predicted clusters (Section 5.4.7).

5.3.5 Experimental determination of kinase binding affinity

From the Surfaceome [189] database, we selected 51 kinases which were available by the KdELECT assay from DiscoveryX. From the ZINC database [10], we selected 4715 compounds purchasable from the Cayman Chemical Company. Using a ConPLex model trained on BindingDB and fine-tuned on DUD-E, we predicted all pairwise interactions between kinases and small molecule drugs. Without previously consulting the literature on kinases or drugs, we selected 5 kinases which were highly represented in the top predictions (*EGFR*, *EPHB1*, *FLT3*, *KIT*, *TGFBR2*). We then selected 19 binding pairs to test, covering 14 drugs with high ConPLex predicted interactions.

We performed K_D determination using the KdELECT assay from the DiscoveryX company, following the procedure from Hie et al. [122]. KdELECT measures competition between test compounds and an immobilized, active-site directed ligand. Ligands are tagged with DNA oligomers, and competition is measured by quantitative polymerase chain reaction (qPCR) of this barcode. BL21-derived *E. coli* were infected with T7 phase strains tagged with each kinase target and incubated with shaking at 32C. Streptavidin-coated magnetic beads were treated with biotinylated ligand at room temperature for 30 minutes, following which the beads were blocked with excess biotin and washed with blocking buffer [SeaBlock (Pierce), 1% bovine serum albumin (BSA), 0.05% Tween 20, 1 mM dithiothreitol (DTT)] to remove unbound ligand. Test compounds were prepared as 111X stocks in 100% DMSO. An 11-point, 3-fold compound diluation series was created, with a top test compound concentration of 10,000 nM. 3 DMSO control points were also used. Test compounds are distributed by acoustic transfer (non-contact dispensing) in 100% DMSO, then diluted into the assays for a final DMSO concentration of 0.9%.

Kinases, ligand-bound affinity beads, and test compounds were combined in 1X binding buffer [20% SeaBlock, 0.17X phosphate-buffered saline (PBS), 0.05% Tween 20, 6 mM DTT] in a 384-well plate, with a final volume of 0.02mL for each reaction. Plates were incubated

for one hour at room temperature with shaking. Affinity beads were washed with wash buffer (1x PBS, 0.05% Tween 20), re-suspended in elution buffer (1x PBS, 0.05% Tween 20, 0.5 mM non-biotinylated affinity ligand), and incubeted for 30 minutes at room temperature with shaking. The concentratino of kinases was measured using qPCR. To compute K_D of the binding, a standard dose-response curve was fit to the Hill equation curves using the Levenberg-Marquardt algorithm (Hill slope = -1).

5.3.6 Genome wide ChEMBL scan

We trained a ConPLex model using BindingDB and DUD-E, and used it to make predictions for all pairs of human proteins against all drugs in ChEMBL. Human protein sequences were taken from the STRING database and processed following [11], resulting in 15,816 proteins between 50 and 800 amino acids long. Small molecule structures were downloaded from ChEMBL 30 [190], resulting in 1,533,652 compounds. Prediction took just under a day, accounting for embedding time.

5.4 Results

5.4.1 Model Overview

To achieve both generalizability and specificity, ConPLex leverages advances in both protein language modeling and metric learning. We start with pre-trained representations and learn a non-linear projection of these representations to a shared space ($\mathbb{R}^{d_h}$). We guide the learning by alternating between two objectives over multiple iterations: a coarse-grained objective of accurately classifying DTIs, and a fine-grained objective of distinguishing decoys from drugs. The coarse-grained objective is evaluated over a low-coverage data set, which trains the model to distinguish between broad classes of drug and target, and makes initial predictions in the right "neighborhood" of the DTI space. The fine-grained objective is evaluated over a

high-coverage data set, which fine-tunes the model to distinguish between true and false positive interactions in the same "neighborhood" and achieve high specificity within a class.

To featurize the inputs, here we use the Morgan fingerprint [210] for small molecules, and embeddings from a pre-trained ProtBert model [209] for proteins. We investigate other choices for features, including several other foundation PLMs in Section C.2.1. We note that our framework is flexible to different methods of featurization, and make recommendations on the selection of informative representations in Section 5.5.

5.4.2 State-of-the-art performance on low-coverage interactions

A key advance of ConPLex is the use of pre-trained protein language models (PLMs) for protein representation. As foreshadowed by Scaiewicz and Levitt [218], PLMs have repeatedly been shown to encode evolutionary and structural information [48], [53], [219], and to enable broad generalization in low-coverage scenarios [11], [12]. Here, we show that ConPLex achieves state-of-the-art performance on three low-coverage benchmark data sets – **BIOSNAP**, **BindingDB**, and **DAVIS** – where it is important to learn the broad strokes of the DTI landscape. In Table 5.2 we show the average area under the precision-recall curve (AUPR) over 3 random initializations of each model evaluated on a held-out test set (Section 5.3.3). Here, we compare with several methods which use non-PLM protein features: MolTrans [185], GNN-CPI [217], and DeepConv-DTI [184]. In addition, we compare to the EnzPred-CPI model from Goldman et al. [197] (developed simultaneously and independently), which uses a PLM for protein featurization but does not perform a co-embedding or utilize a contrastive training step. Finally, we compare with the single-task Ridge regression model described in [197], which trains a different model per-drug rather than a single model for the entire benchmark.

Observing the strength of ConPLex to generalize on low-coverage data, we sought to evaluate its performance on fully zero-shot prediction. **Unseen drugs** and **Unseen targets** are variants of the BIOSNAP data set where drugs/targets in the test set do not appear in

any interactions in the training set (Section 5.3.2). Note that for the unseen drugs setting, the Ridge model cannot be applied since a different model must be trained for each drug that appears in the training set. We show that ConPLex achieves the best zero-shot prediction performance (Table 5.2), further demonstrating the applicability of the model to large-scale, very low-coverage prediction tasks.

5.4.3 Contrastive learning enables high-specificity DTI mapping

Another key advance of our method is the use of contrastive learning to fine-tune model predictions on high-coverage data to achieve high specificity. Recently, Heinzinger et al. [220] demonstrated the use of semi-supervised contrastive learning for effective protein embedding-based annotation transfer. Here, we adapt contrastive learning to a fully-supervised setting and demonstrate that the contrastive training is essential to achieving specificity using DTI pairs from the Database of Useful Decoys (**DUD-E**) [207]. The DUD-E data set contains 57 protein targets and drugs which are known to interact with each target. However, it also contains 50 negative "decoy" small molecules for each drug, which have similar physicochemical properties to the truly interacting small molecule, but are known to not bind the target. Thus, accurate prediction on DUD-E requires a model to achieve high-specificity and to accurately differentiate between highly similar compounds. Additionally, DUD-E contains four different classes of targets — G-protein coupled receptors (GPCRs), kinases, proteases, and nucleases — so models must generalize across target classes (note that single task models don't have this generalization requirement, since a different model is trained per-target).

We derive evaluation sets from DUD-E by holding out 50% of proteins in each target class for testing and using the remaining targets for training (full splits are specified in Section C.1). Here, we evaluate a ConPLex model trained on BIOSNAP, both with and without contrastive training on DUD-E, and show that contrastive training is essential to achieving specificity on decoys.

For each target in the DUD-E test set, we use t-SNE to visualize the target alongside all

drugs and decoys using embeddings learned by both versions of the model. Figure 5.3a,b shows one such example, the tyrosine kinase *VGFR2*. We also show the distribution of distances in the latent space between the target embedding and the embeddings of the drugs and decoys for each model (Figure 5.3c, d) (p-values from one-sided t-test). Without contrastive training, drugs are interspersed with decoys and are far away in space from the target, while ConPLex clusters most true drugs very close to both each other and the *VGFR2* embedding.

In Figure 5.3e, we show a quantitative analysis of all 31 test-set targets. We compute the effect size (Cohen's d) of the difference between predicted drug and decoy scores. We plot these effect sizes for ConPLex trained with and without contrastive training. An increase in the effect size indicates that the co-embedding distances learned by the model better represent binding specificity. The effect size increases for every target, and the median effect size between predicted true and decoy compound scores was 0.730 prior to contrastive training compared to 4.716 after. For each class of targets, we also report the median p-value (one-sided t-test) between drug and decoy scores predicted by ConPLex. While contrastive training has an extremely large impact on specificity in high-coverage domains, we also show that this additional training does not significantly decrease the model performance on low-coverage benchmarks via an ablation study in Section 5.4.11.

In addition to evaluation on DUD-E, we also evaluate ConPLex on five benchmark data sets derived from family-specific enzyme-substrate screens (Section 5.3.2). These data sets are extremely high coverage, generally including data points for all possible pairs of drugs and targets. We find that in this regime, ConPLex and other PLM based models like EnzPred-CPI have strong but highly variable performance, and are still generally outperformed by a Ridge regression model (Section 5.4.10) as shown previously in [197]. However, a fine scale single-task model is limited in its generalizability beyond the enzyme family on which it was trained (Section 5.5).

5.4.4 Discovering DTIs with sub-nanomolar binding affinity

Since ConPLex exhibited strong performance on several benchmark data sets, we next sought to experimentally validate predictions using an *in vitro* biochemical binding assay. We selected 51 kinases from the Surfaceome database [189] with commercially-available assays from the DiscoveryX company, and used ConPLex to scan against a set of 4715 compounds from the ZINC database [10] purchasable from the Cayman Chemical Company (Section 5.3.2). We selected 19 interactions spanning 5 kinases and 14 compounds in an unbiased manner (these pairs were chosen based solely on top scoring ConPLex predictions, without any use of prior knowledge from experimental results or in the literature). We determined K_D values for each of the 19 interactions (Table 5.3), finding that 12/19 pairs tested had K_D values less than 100nM. Of these, four bound with sub-nanomolar affinity, all of which recapitulate known interactions in the literature. Weglicki et al. identified AG-1478 as an *EGFR* inhibitor, but noted that its therapeutic use may be limited due to triggering hypomagnesemia and cardiac dysfunction [222]. Sordella et al. described the downstream impact in lung cancer when Gefitinib inhibits *EGFR* [223]. In a review of Nintenanib discovery, Roth et al. noted it as an *FLT3* inhibitor [224], and Wang et al. described Linifanib inhibition of *FLT3* [225].

We also identify an interaction we believe to be novel, between *EPHB1* and PD-166326 with nearly sub-nanomolar affinity ($K_D = 1.30$). Wolff et al. previously identified PD-166326 as a tyrosine-kinase inhibitor [226], but did not report any binding to *EPHB1*, and DrugBank [9] only lists *ABL1* as a known target (DrugBank ID: DB08339). *EPHB1* has been implicated in chronic pain [227], [228]; at time of publication, there are no known inhibitors of *EPHB1* listed in the Protein Kinase Inhibitor Database (PKIDB) [221], and our findings indicate that PD-166326 may act as a novel binder to *EPHB1*. Future work could involve further characterization of this interaction, its impact on *EPHB1* function, and possible therapeutic outcomes. In Figure 5.4b, we show that PD-166326 is the only compound from our screen close to *EPHB1* in co-embedding space.

Notably, all three of the compounds that we predicted to interact with *TGFBR2* were false positives (Wortmannin, Pluripotin, Monorden). Despite its significance in cancer signaling [229], there are no known inhibitors of *TGFBR2* in PKIDB, suggesting that it may be difficult to target via small-molecule drugs.

5.4.5 Selection of a threshold based on experimental results

Using an experimental binding affinity of <100 nM as a label of a true positive interaction, we constructed a precision-recall curve using ConPLex predicted probability of interaction (AUPR = 0.91). This allows us to understand the calibration of the ConPLex prediction, and set thresholds for future prediction. At a threshold of 0.923, we reach precision = 1.0, indicating that all tested interactions above this threshold truly interacted. We show the curve with all thresholds listed in Figure 5.5.

5.4.6 Incorporating drug binding information improves protein representations

One of the advantages of the co-embedding approach that our model takes is the ability to visualize and investigate the shared embedding space. For instance, we show in Figure 5.4a-c that kinases and their inhibitors tend to co-localize within the space. Seeking to expand our analysis, we subsequently mapped all 2716 predicted surface proteins from the Surfaceome database into ConPLex embedding space, and investigated their representations. In Figure 5.4d, we show the projections all Surfaceome proteins, colored by their classification into one of five functional categories (from Almén et al. [230]) – transporters, receptors, enzymes, miscellaneous, and those that are unclassified. ConPLex projections of surface proteins cluster in embedding space by functional type, with transporters and receptors especially separating from other classes.

However, the Almén functional classification is quite broad and may group proteins with

vastly different functions and binding properties. We further demonstrate the link between ConPLex projections and protein function, by evaluating how the learned DTI embedding space separates proteins by domains contained therein. We identified Pfam domains [231] for each protein in the Surfaceome database using HMMscan [62] and compared the projections of proteins that share the same domains. We identified 780 unique domains across all proteins, of which 126 domains were represented in at least 10 proteins. To quantitatively evaluate the coherence of ConPLex embeddings, we trained separate logistic regression classifiers for each domain to separate proteins with that domain from others, and used the model's confidence ($log(\frac{p}{1-p})$) for in-sample proteins as a measure of separation for the domain. We find that for all 126 domains, the model more confidently discriminated domains when trained on ConPLex representations than the baseline ProtBert embeddings.

Figure 5.4f shows the change in confidence scores for all 126 domains, where the dotted line represents equal confidence using either ConPLex or ProtBert. We find that while prediction of all domains were improved using ConPLex, proteins with kinase domains (PF14575, PF01404, PF00069, PF07714) separated especially well, while 7-transmembrane (7TM) domains characteristic of GPCRs (PF00001, PF00002, PF00003, PF13853) showed more modest improvement. In Figure 5.4e, we show the same visualization of projections as in Figure 5.4d, but colored by another top-differentiated domain, PKinase (PF00069). As discussed previously, ConPLex was trained contrastively with several kinase targets and excels at kinase prediction on DUD-E (Figure 5.3e), so it is unsuprising that proteins with these domains separate well. In fact, one of the top differentiated domains is the Ephrin ligand binding domain (PF01404), which is responsible for binding to the ephrin ligand [232]. While the model was also trained contrastively with 7TM GPCR targets, there was much less training data available. In addition to the dearth of training examples, GPCRs are less soluble than kinases and tend to exhibit more dynamic behavior – all of which contribute to difficulty predicting ligand binding. Future work in this area might adjust distances in this landscape to account for the low metric entropy of biological sequences, as demonstrated in

Berger, Waterman, and Yu [233].

5.4.7 Functional community detection in the Surfaceome

Individual domain performance To evaluate the extent to which ConPLex embeddings could be used to separate proteins by function in embedding space, we trained a logistic regression classifier on a binary classification for each domain, using either ConPLex or ProtBert embeddings as features. We quantify the quality of separation by the model confidence, computed as the mean $log(\frac{p}{1-p})$ for in-class proteins, where p is the logistic classifier predicted probability. In Table 5.4, we report the model confidence using either ConPLex or ProtBert embeddings for kinase domains, cadherin domains, and 7TM domains. ConPLex improves over ProtBert for all domain families, but performs especially well on kinases, while performing relatively weaker on cadherin and 7TM proteins. We show a histogram of results on all Pfam domains tested in Figure 5.6

Spectral Clustering In addition to measuring the quality of the latent space using a classifier, we also tried an unsupervised approach. We trained a spectral clustering algorithm with $k = 10$ clusters. Then, for each domain, we compute the adjusted mutual information (AMI) between the clustering assignments and the presence/absence of that domain. A higher AMI means that proteins with the same domain are more likely to share a cluster. We find that ConPLex embeddings result in signficantly higher AMIs than ProtBert (paired t-test $p = 5.37e - 4$). We show clustering results for each domain as well as summary histograms in Figure 5.7. We also show the distribution of domain cluster sizes for all 780 domains identified in at least one protein, and our cutoff that a domain must have > 10 proteins for us to consider it in Figure 5.7D.

5.4.8 Adapting ConPLex for affinity prediction

While we have to this point been using the model to predict probabilities of interaction and perform binary classification, we show that ConPLex can be easily adapted to perform binding affinity prediction, and that this model too achieves state-of-the-art performance. The final step of our binary interaction predictor is converting the cosine distance between the projections in the DTI space to a probability using a sigmoid activation (Section 5.3.3). However, it is completely natural to replace this activation with a dot product between the two projections, which enables the model to make real-valued predictions, which can then be interpreted as a binding affinity. We evaluated ConPLex trained for affinity prediction on the Therapeautics Data Commons (TDC) DTI Domain Generalization (**TDC-DG**) benchmark. The TDC-DG benchmark contains binding affinity (IC_{50}) data from interactions patented between 2013 and 2018, with the test set drawn from interactions patented in 2019-2021 (Section 5.3.2). Thus, this data requires out-of-domain generalization and corresponds with the real-life scenario of training on interactions up to a known point, and predicting interactions which are yet to be documented. We trained ConPLex to predict binding affinity with 5 random train/validation splits, and achieve an average Pearson correlation coefficient between the true and predicted affinity of 0.538($\pm$0.008) on the held-out test set. At submission, ConPLex is the top performing method on the TDC-DG benchmark on TDC (Table 5.5).

To investigate the strengths and limitations of ConPLex for affinity prediction, we evaluated performance by target type. Targets were annotated with Pfam domains [231] using HMMscan [62], and the Pearson correlation (PCC) between predicted and true IC_{50} was computed over targets in each family (Table 5.6). We observed especially strong performance on 12 immunoglobulin targets (Pfam domains PF13927, PF13895, PF07679, PF00047), where we observed a PCC of 0.803. In keeping with our previous finding of ConPLex's relative strength on kinases over GPCRs (Figure 5.3e), we observed a correlation of 0.578 on 94 protein targets with PKinase domains (PF07714, PF00069), including targets with

SH3 domains (PF00018, PF07653; 13 targets; PCC = 0.705) and PI3K domains (PF00613, PF00792; 6 targets; PCC = 0.633). However, we observed substantially weaker performance on 7TM domains (PF00001; 14 targets; PCC = 0.254) and GPCR domains (PF10320; 8 targets; PCC = 0.176). To assess ConPLex's variability in its accuracy, we computed a 95% prediction interval based on a linear regression between the true and predicted IC_{50}. While the correlations were strong, we found substantial variability around the true IC_{50}, with the width of the prediction interval around the true $ln(IC_{50})$ being ± 4.89 $ln(nM)$ (Figure 5.8). Altogether, the variability in ConPLex's performance across domains makes it important to understand the target of interest when using ConPLex to predict binding affinity.

5.4.9 Error limits of binding affinity predictions

ConPLex achieves the strongest performance on the TDC DTI-DG benchmark for affinity prediction. However, when predicting binding affinity, is important to consider the error limits and range of applicability of the method. In , we show the true vs. predicted $ln(IC_{50})$ for all drug-target interactions in the data set. We compute the kernel density estimate for the predictions, and a 95% prediction interval, which shows the prediction error for the test set.

5.4.10 Results on EnzPred data sets

Results of ConPLex, EnzPred-CPI, and a single-task ridge regression model on several protein-family specific data sets compiled by Goldman et al. [197]. These include β-ketoacid cleavage (**BKACE**) [206], **Esterase** [203], **Glycosyltransferases** [204], **Halogenase** [205], and **Phosphatase** [202] enzymes. Following [197], we performed a 10-fold cross validation where the data were split into train-test sets by target, so that all drugs appear in both the training and test set, but no target does. We report the average of 3 random initializations in Table 5.7. "Average AUPR" means AUPR was computed per-drug then averaged.

5.4.11 Margin decay ablation and optimization

We evaluated the impact of various different margin decay schemes, as well as the impact of contrastive learning overall on model performance. We report the average AUPR over 3 random initializations on the BindingDB data set. The baseline without contrastive learning has the highest AUPR, but as we have shown contrastive learning is essential for specificity on fine-scale data sets. The best performing model with contrastive learning is within 5% of the performance without. In addition to the Tanh decay with restart we describe in the main text, we evaluate the performance of a Tanh decay without a restart, a cosine decay with and without restarts, and a constant margin (no decay). We find that the Tanh decay with restart performs the best, and show the full results in Table 5.8.

5.5 Chapter Perspectives

Much previous work has recognized the value of meaningful drug representations [195], [234] for DTI prediction, yet relatively little work has focused on the target protein representation. As the first method to use pre-trained protein language models (PLMs) for DTI prediction, ConPLex is yet another example of the power of transferring learned representations for biology [11], [185], [219], [235], [236]. This approach enables broad generalization to unseen proteins, as well as extremely fast model inference (>10x speed-up even over other sequence-based approaches, Section C.3). This speed is particularly valuable for drug re-purposing and iterative screening, where large compound libraries are evaluated against hitherto-uncharacterized proteins implicated in a disease of interest. The co-embedding approach which enables this speedup could also be effective for integrative multi-structure models (e.g., the IMP framework [237]) where efficient scanning of possible combinations is important. Recent methods have also demonstrated the power of PLMs for transferring knowledge between species [11], and our framework may enable more accurate transfer of DTI from the model organisms on which drugs are initially tested, to their eventual use in human

patients. Skolnick and Zhou [238] have reported the importance of considering small molecule binding pockets for protein-protein interaction prediction; thus our DTI-informed protein representations may also be useful in that context. While structural similarity is often implicitly learned by PLMs, future work could explicitly incorporate structure where such data is available, perhaps by incorporating a more advanced projection architecture like the Geoformer [146].

It has been shown in previous work that the performance of different PLMs vary on different tasks, and that there is not one clearly "best" language model [186], [239], [240]. While we have chosen to use ProtBert here, it is likely that other existing or newly developed language models may yield better performance for certain types of drugs or targets. Likewise, advancements in drug representation may improve performance— the ConPLex framework is flexible to different input features, and it remains important to experiment with different feature choices for the task at hand (Section C.2).

ConPLex approaches the DTI decoy problem from the perspective of adversarial machine learning, where the model must act as a discriminator for adversarial examples from the decoy database. This approach is directly enabled by the co-embedding architecture— to compute the triplet distance loss, the protein and drugs must be co-embedded, and the distance between them must be meaningful and simply computed. Such an approach would not be feasible using a model which concatenates features up front, nor for a model which has significant computation defining the probability of interaction after the co-embedding. Thus the shared lexicographic space in which we embed the proteins, targets, and decoys is key. Future work could explore adapting molecular generation methods such as JT-VAE or HierG2G [241], [242] to directly act as a generator for decoys. High-specificity DTI prediction is valuable beyond decoy detection— greater specificity of inference can help improve personalized medicine or the modeling of drug effects against rare variants from under-represented populations.

It is also important to consider the coverage of the problem to select an appropriate

method. While we recommend the use of PLM-based features in all cases, if enough data is available, for specific enzyme-family prediction tasks we still recommend the use of single-task models [197]. To verify individual interactions, energy-based molecular docking will likely be more accurate, although at the cost of being substantially slower [177]. Different classes of computational tools for DTI prediction each have varying strengths, and the highest quality predictions can be achieved by leveraging all of these methods together where each is most fit.

Drug discovery is a fundamental task for human health, yet remains both extremely expensive and time-consuming, with the median drug requiring over 1 billion dollars [243] and 10 years [244] from development to approval and distribution. While experimental results will remain the gold-standard for validating drug functionality, *in silico* prediction of drug-target binding remains much faster and cheaper and so will continue to play an important role in early screening of therapeutic candidates [245]. To address this step in the drug design pipeline, we have introduced ConPLex. DTI prediction methods should be able to generalize to unseen types of drugs and targets, while also discriminating between highly similar molecules with different binding properties. ConPLex tackles both of these challenges through its dual use of protein language models and contrastive learning. We hope that its broad applicability, specificity on decoys, and ability to scale to massive data will allow ConPLex to be a critical step in this pipeline and contribute to the efficient discovery of effective therapeutics.

Table 5.1: Full specification of benchmark data sets. We report the number of unique drugs and targets, the median (drug/target) coverage, and the number of training, validation, and test samples in each data set. Number of pairs are shown as (positive/negative), except for TDC-DG [191], [198], which is a regression task, thus total number of pairs is shown. We consider BIOSNAP [199], BindingDB [200], DAVIS [201], and TDC-DG as low-coverage, while Phosphatase [202], Esterase [203], Glycosyltransferase [204], Halogenase [205], BKACE [206], and DUD-E [207] are considered high-coverage. † Because DUD-E is a decoy data set, we report as coverage the median number of true drugs or decoys for each target.

Data Set	Drugs	Targets	Median Coverage	# Training	# Validation	# Test
BIOSNAP	4510	2181	0.0023 / 0.0020	9670 / 9568	1396 / 1352	2770 / 2727
Unseen Drugs				9535 / 9616	1383 / 1353	2918 / 2675
Unseen Targets				9876 / 9499	1382 / 1386	2578 / 2762
BindingDB	7165	1254	0.0008 / 0.0010	6334 / 6334	927 / 5717	1905 / 11384
DAVIS	68	379	0.3707 / 0.3676	1043 / 1043	160 / 2846	303 / 5708
TDC-DG	140746	477	0.0021 / 0.0005	146891	36539	49028
Phosphatase	165	218	1.0 / 1.0	5054 / 27286	—	370 / 3260
Esterase	96	146	1.0 / 1.0	2150 / 10426	—	926 / 514
Glycosyltransferase	89	54	0.9259 / 0.9778	725 / 3042	—	113 / 417
Halogenase	62	42	1.0 / 1.0	303 / 1991	—	20 / 290
BKACE	17	161	1.0 / 1.0	255 / 2193	—	19 / 270
DUD-E †				8996 / 406208	—	11430 / 521132
GPCR	99671	5	18563			
Kinase	315399	26	15409			
Protease	286089	15	9271			
Nuclear	151133	11	16257			

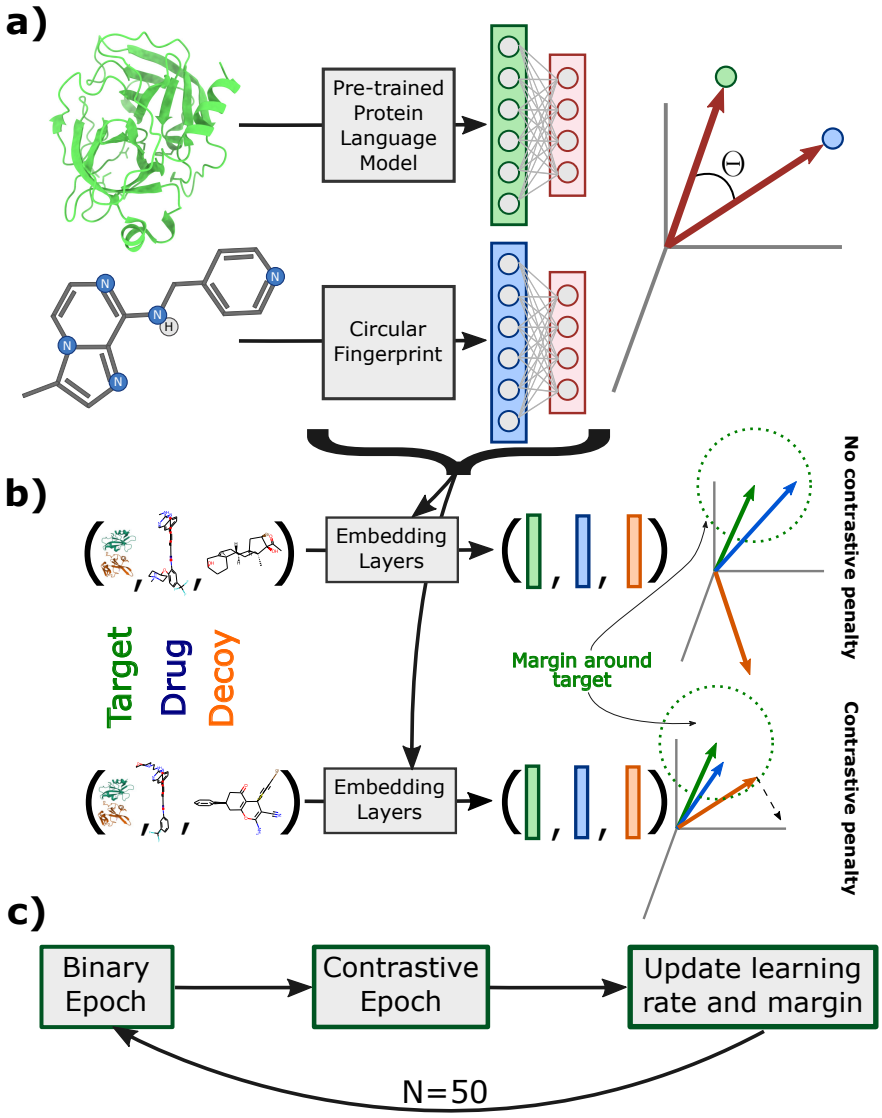


Figure 5.2: **Outline of the ConPLex model architecture and training framework.** ConPLex is trained in two phases, to optimize both generalizability and specificity. **(a)** Protein features are generated using a pre-trained protein language model (here ProtBert [209]), and drug features are generated using the Morgan fingerprint [210]. These features are transformed into a shared latent space by a learned non-linear projection. The prediction of interaction is based on the cosine distance in this space, and the parameters of the transformation are updated using the binary cross-entropy on a low-coverage data set. **(b)** In the contrastive phase, triplets of a target, drug, and decoy are transformed in the same way into the shared space. Here, the transformation is treated as a metric learning problem. Parameters are updated using the triplet distance loss on a high-coverage data set to minimize the target-drug distance while maximizing the target-decoy distance. No additional penalty is applied if the target-decoy distance is greater than the target-drug distance plus some margin. **(c)** ConPLex is trained in alternating epochs of the binary and contrastive phase to simultaneously optimize both objectives. After each round, learning rates and the contrastive margin are updated according to an annealing scheme.

Table 5.2: **ConPLex is highly accurate and generalizes broadly in low coverage settings.** ConPLex outperforms several state-of-the-art methods, including EnzPred-CPI [197], MolTrans [185], GNN-CPI [217], and DeepConv-DTI [184], as well as a simple single-target Ridge regression model, on several low- and zero- coverage benchmark data sets. We report the average and standard deviation of the area under the precision-recall curve (AUPR) for 5 random initializations of each model. Metrics for models with † are taken from [185]. Ridge regression cannot be applied for the **Unseen Drugs** data set, since a separate model is trained for each drug in the training set.

Data Set	ConPLex	EnzPred-CPI	MolTrans	GNN-CPI†	DeepConv-DTI†	Ridge
BIOSNAP	$\mathbf{0.897 \pm 0.001}$	0.866 ± 0.003	0.885 ± 0.005	0.890 ± 0.004	0.889 ± 0.005	0.641 ± 0.000
BindingDB	$\mathbf{0.628 \pm 0.012}$	0.602 ± 0.006	0.598 ± 0.013	0.578 ± 0.015	0.611 ± 0.015	0.516 ± 0.000
DAVIS	$\mathbf{0.458 \pm 0.016}$	0.277 ± 0.009	0.335 ± 0.017	0.269 ± 0.020	0.299 ± 0.039	0.320 ± 0.000
Unseen Drugs	$\mathbf{0.874 \pm 0.002}$	0.844 ± 0.005	0.863 ± 0.005	-	0.847 ± 0.009	N/A
Unseen Targets	$\mathbf{0.842 \pm 0.006}$	0.795 ± 0.004	0.668 ± 0.045	-	0.766 ± 0.022	0.617 ± 0.000

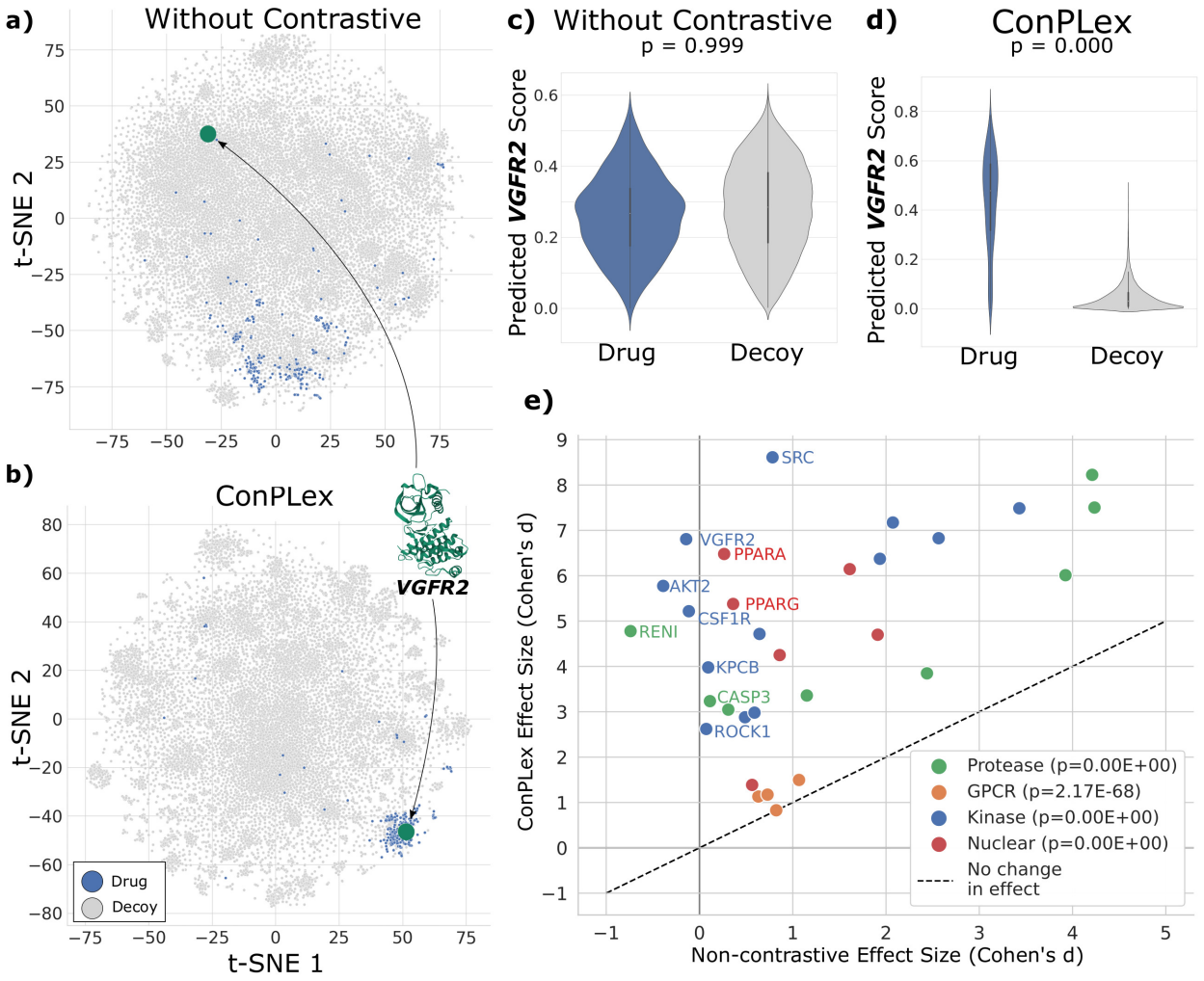


Figure 5.3: **Contrastive training enables high-specificity in discriminating drugs from decoys.** We demonstrate that contrastive learning is essential for ConPLex to achieve high specificity using the DUD-E [207] data set of drugs and decoys (non-binding small molecules with similar physicochemical properties to the true drugs). **(a, b)** Using t-SNE, we show the learned ConPLex latent space for *VGFR2* (green) and known drugs (blue) and decoys (grey). Without contrastive training, drugs and decoys representations do not separate and true drugs are far from their target. With contrastive training, *VGFR2* and drugs cluster very tightly compared to decoys. **(c, d)** ConPLex predictions significantly differentiate between drugs and decoys after contrastive training ($p = 0.000$ paired t-test), but do not differ at all without such training ($p = 0.999$). **(e)** We compute the effect size between drug and decoy predictions using Cohen's d for all 31 targets in the test set. Targets are classified as proteases (green), GPCRs (orange), kinases (blue), and nuclear proteins (red). This effect is computed for ConPLex both with and without contrastive training. Contrastive training increases the effect size for every target (median 0.730 vs 4.716). For each class, we report the median $p-$value for ConPLex drug vs. decoy predictions. ConPLex performs particularly well for kinases and nuclear proteins, and more poorly for GPCRs.

Table 5.3: We selected and tested 19 potential binding interactions, where the selection of tests was done based solely on ConPLex predicted interaction, and without consulting previous experiments or literature. We determined the K_D values for each interaction via an *in vitro* biochemical assay (Section 5.3.5), and we show here the K_D in nM units. Twelve exhibited binding affinity in the nanomolar range, including four (denoted with *****) binding with sub-nanomolar affinity. The only target for which we incorrectly predicted there would hits was *TGFBR2*, which has no known inhibitors in PKIDB [221], suggesting it may be difficult to target. We recapitulate several known interactions (**Results**), and find a tightly-binding interaction between *EPHB1* and *PD 166326* (**bold**) which to our knowledge has not been previously characterized.

	EGFR	**EPHB1**	**FLT3**	**KIT**	**TGFBR2**
AG-1478	0.33*	-	-	-	-
Gefitinib	0.60*	-	-	-	-
Janex 1	26.00	-	-	-	-
SB-431542	>1e4	-	-	-	-
AG-1296	>1e4	-	62.00	27.00	-
ZM 447439	>1e4	-	-	>1e4	-
PD-166326	-	**1.30**	-	-	-
Nintendanib	-	-	0.17*	-	-
Linifanib	-	-	0.72*	1.70	-
Sorafenib	-	-	7.20	36.00	-
Imatinib	-	-	-	6.00	-
Wortmannin	-	-	-	-	>1e4
Pluripotin	-	-	-	-	>1e4
Monorden	-	-	-	-	>1e4

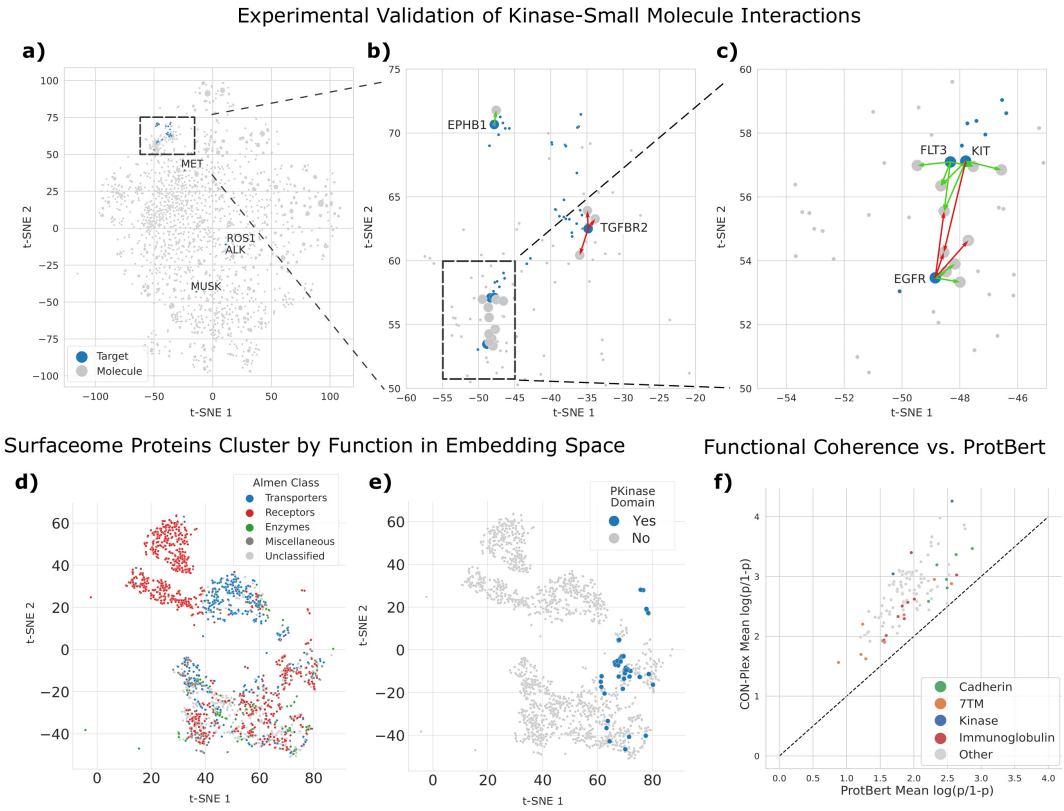


Figure 5.4: **The shared representation space learned by ConPLex captures drug-target interaction and protein function.** **(a)** We show that 51 kinases from the Surfaceome [189] database cluster together in ConPLex embedding space, but occupy just a small section of the entire space when co-embedded with the compounds from the ZINC [10] Cayman-purchasable library. **(b, c)** Zooming in on the full embedding space highlights drug-target pairs chose for experimental validation. *EPHB1* has only a single compound nearby in embedding space, PD-166326, which was confirmed to bind with single-digit nanomolar affinity. *FLT3* and *KIT* are neighbors in embedding space, and tightly bind many of the same compounds; both bind to Linifinib with $< 2nM$ affinity. *EGFR* was not found to bind to any of the compounds also tested with *FLT3* and *KIT*, but binds three other drugs nearby in the embedding space, two of which bind with sub-nanomolar affinity. One the other hand, none of the three compounds we tested nearby *TGFBR2* (Wortmannin, Pluripotin, Monorden) were found to bind. **(d)** ConPLex representations of all cell surface proteins from the Surfaceome [189] cluster by functional class as assigned in Almén et al. [230]. **(e)** These representations also cluster by several functional Pfam domains [231], such as the PKinase domain (PF00069) shown in blue. **(f)** We evaluated the coherence of representations for each domain by training a logistic regression classifier and report the model's average confidence for proteins containing that domain as $log(\frac{p}{1-p})$. ConPLex separates all 126 domains better than the un-transformed ProtBert embeddings ($p = 4.85 \times 10^{-54}$, paired t-test), discriminating kinase domains (blue) especially well. We have also highlighted other classes of domains, including cadherins (green), 7-transmembrane proteins (orange), and immunoglobulins (red).

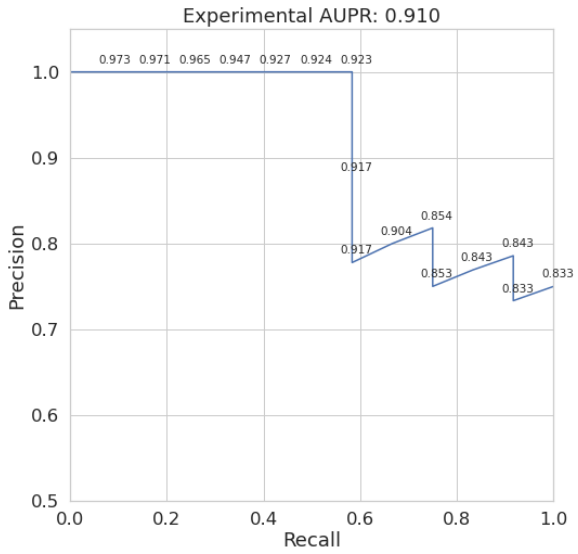


Figure 5.5: **Precision-recall curve computed on experimentally tested interactions**

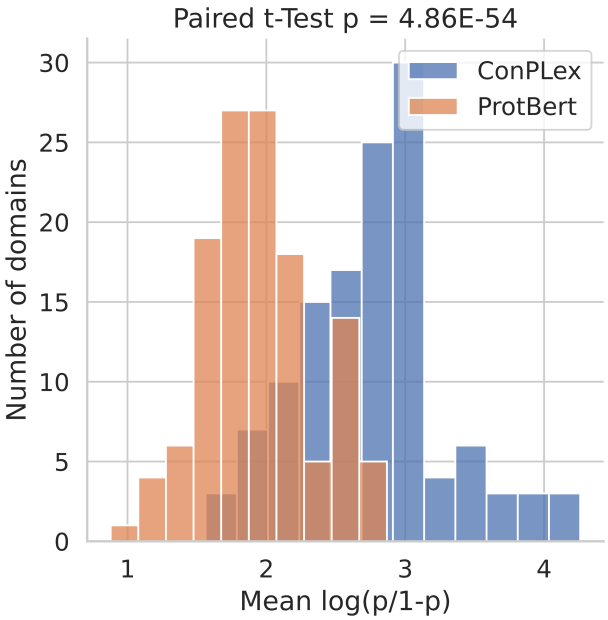


Figure 5.6: Distribution of domain separation for all 126 domains represented in ≥ 10 proteins ($p = 4.85 \times 10^{-54}$, paired t-test).

Table 5.4: We evaluated the coherence of representations for each domain by training a logistic regression classifier and report the model's average confidence for proteins containing that domain as $log(\frac{p}{1-p})$. While ConPLex separates all domains better than the untransformed ProtBert embeddings, it achieves especially strong performance on kinases.

Domain Accession	**Domain Name**	**ConPLex** $log(\frac{p}{1-p})$	**ProtBert** $log(\frac{p}{1-p})$
PF14575.7	EphA2_TM	**4.26**	2.57
PF01404.20	Ephrin_lbd	**4.26**	2.57
PF00069.26	Pkinase	**3.04**	1.69
PF07714.18	PK_Tyr_Ser-Thr	**3.04**	1.69
PF01833.25	TIG	**3.40**	1.96
PF18452.2	Ig_6	**2.51**	1.83
PF16706.6	Izumo-Ig	**2.57**	1.91
PF17736.2	Ig_C17orf99	**2.62**	2.01
PF16680.6	Ig_4	**2.33**	1.76
PF16681.6	Ig_5	**2.29**	1.86
PF13895.7	Ig_2	**2.01**	1.59
PF10613.10	Lig_chan-Glu_bd	**3.03**	2.64
PF00060.27	Lig_chan	**3.03**	2.64
PF13927.7	Ig_3	**1.94**	1.56
PF00047.26	ig	**1.92**	1.54
PF08758.12	Cadherin_pro	**3.19**	2.34
PF16492.6	Cadherin_C_2	**3.36**	2.63
PF15974.6	Cadherin_tail	**3.47**	2.87
PF01049.18	Cadherin_C	**2.95**	2.48
PF16184.6	Cadherin_3	**2.58**	2.22
PF00028.18	Cadherin	**2.81**	2.49
PF08266.13	Cadherin_2	**2.88**	2.56
PF00001.22	7tm_1	**2.95**	2.30
PF00002.25	7tm_2	**1.56**	0.88
PF00003.23	7tm_3	**2.20**	1.24
PF13853.7	7tm_4	**2.88**	2.56

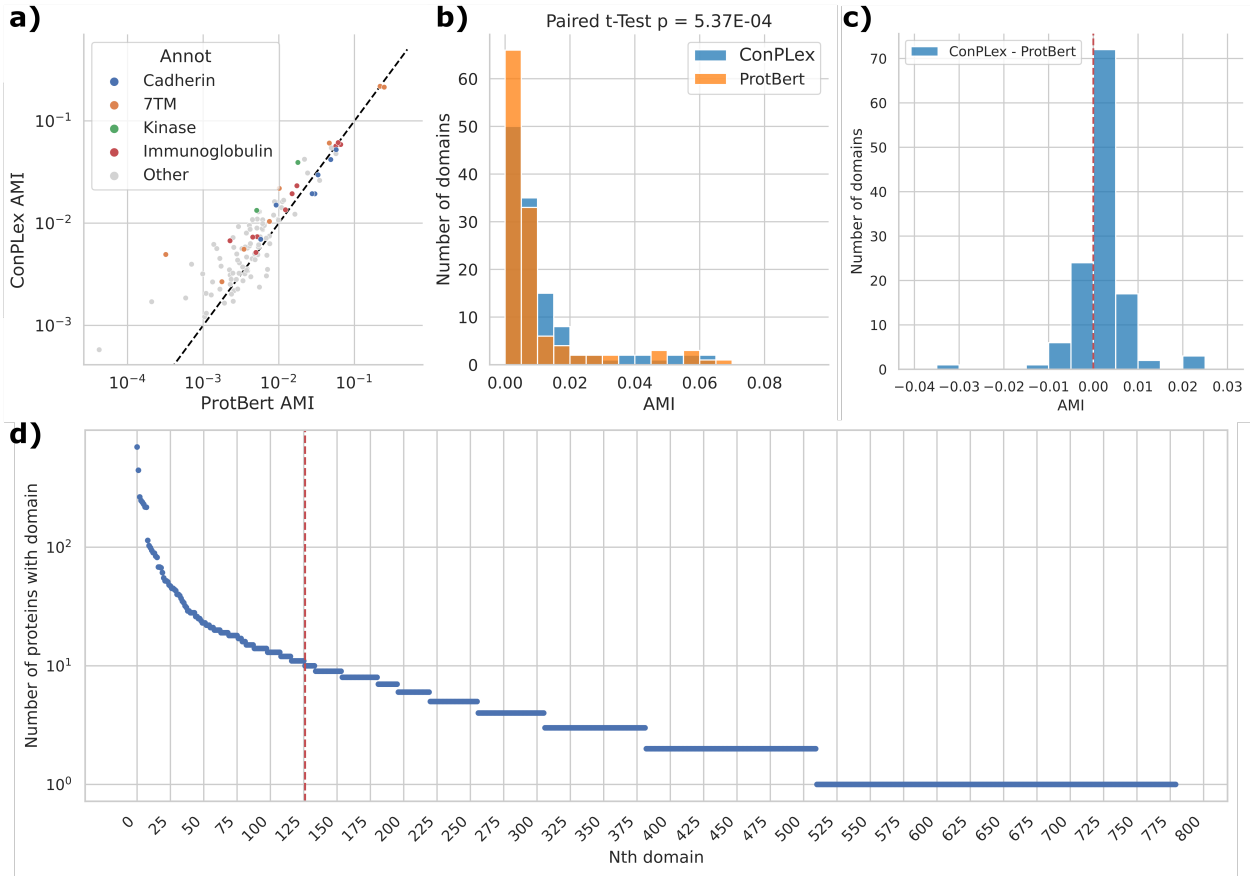


Figure 5.7: **Analysis of domain coherence on Surfaceome (a)** ProtBert vs. ConPLex spectral clustering AMI to true separation for each domain. Above the diagonal dotted line means that ConPLex improves upon ProtBert representations. **(b)** Histogram of clustering quality AMI for ConPLex and ProtBert **(c)** Histogram of residuals. To the left of 0, ProtBert performed better, while ConPLex performed better for domains to the right of 0. **(d)** For all domains that were annotated using HMMscan, we plot the number of proteins that have that domain. We don't evaluate any domains that appear in fewer than 10 proteins, since clustering or classification metrics could be subject to noise. As a result, we evaluate on 126 domains.

Table 5.5: **ConPLex can be adapted for state-of-the-art affinity prediction.** By replacing the cosine distance in the final step of ConPLex with a dot product between the projections, ConPLex can be used for affinity prediction rather than binary classification. The Therapeatics Data Commons-Domain Generalization (**TDC-DG**) data set contains IC_{50} values for patented drug-target pairs, where training/testing data are split from before/after 2018. We report the average and standard deviation of the Pearson Correlation Coefficient between true and predicted values across five train/validation splits. Metrics for all methods other than ConPLex come from the TDC leaderboard [191], [198], where at the time of submission ConPLex is the best performing method.

Model	PCC
ConPLex	**0.538 ± 0.008**
MMD	0.433 ± 0.010
CORAL	0.432 ± 0.010
ERM	0.427 ± 0.012
MTL	0.425 ± 0.010
GroupDRO	0.384 ± 0.006
AndMASK	0.288 ± 0.019
IRM	0.284 ± 0.021

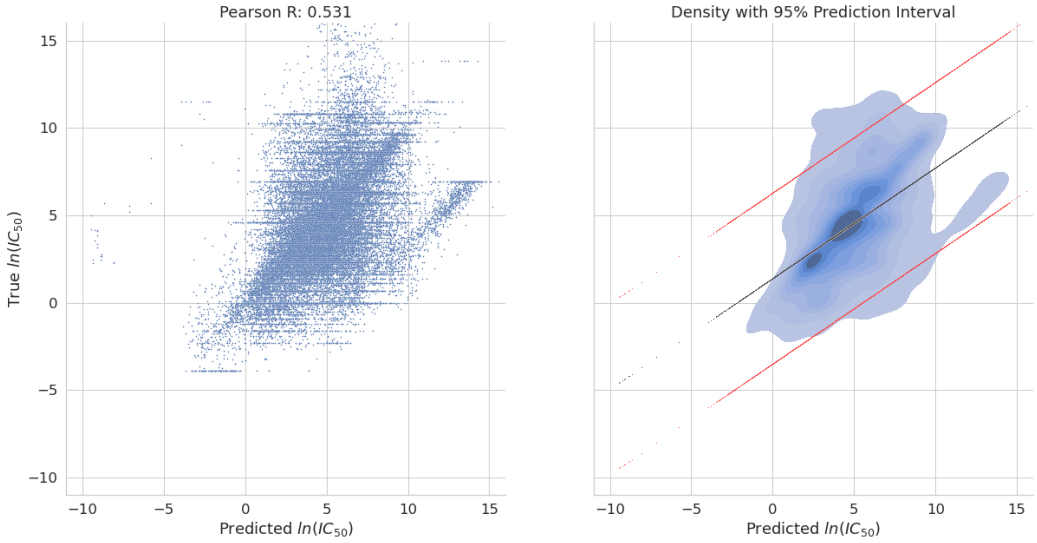


Figure 5.8: **Error limitations of affinity prediction.** **(a)** Scatterplot comparing the predicted vs. true affinity on the TDC DTI-DG [191] benchmark data set ($ln(IC_{50})$). ConPLex achieves a Pearson correlation of 0.531. **(b)** KDE plot of predictions, with a 95% prediction interval computed. While the overall correaltion is strong, the error for any individual prediction is still quite high, and true affinity may vary over several orders of magnitude at the same level of predicted affinity.

Table 5.6: **Breakdown of affinity accuracy by PFam domain.**

Domain Accession	Domain Name	Unique Targets	Pearson R	Pearson P
PF00041.22	fn3	7	0.9554	5.6999e-241
PF13927.7	Ig_3	12	0.8028	0.0
PF13895.7	Ig_2	12	0.8028	0.0
PF07679.17	I-set	12	0.8028	0.0
PF00047.26	ig	12	0.8028	0.0
PF07686.18	V-set	8	0.7747	0.0
PF00018.29	SH3_1	13	0.7049	0.0
PF07653.18	SH3_2	13	0.7049	0.0
PF14604.7	SH3_9	12	0.7044	0.0
PF00850.20	Hist_deacetyl	11	0.6679	2.4276e-183
PF06293.15	Kdo	11	0.6670	1.4394e-176
PF00454.28	PI3_PI4_kinase	9	0.6526	0.0
PF00613.21	PI3Ka	6	0.6335	0.0
PF00792.25	PI3K_C2	6	0.6335	0.0
PF00169.30	PH	6	0.6240	1.73e-321
PF14531.7	Kinase-like	10	0.6148	0.0
PF07714.18	PK_Tyr_Ser-Thr	94	0.5785	0.0
PF00069.26	Pkinase	94	0.5785	0.0
PF00017.25	SH2	18	0.5012	0.0
PF00001.22	7tm_1	14	0.2544	9.8827e-32
PF10320.10	7TM_GPCR_Srsx	8	0.1757	6.7301e-12

Table 5.7: Comparison of PLM-based drug-target interaction models with a per-drug Ridge regression model on enzyme-family specificity benchmark data sets.

Benchmark	Model	Average AUPR
BKACE	ConPLex	0.53 ± 0.04
	EnzPred-CPI	0.41 ± 0.01
	Ridge (Single-Task)	**0.62 ± 0.02**
Esterase	ConPLex	0.46 ± 0.02
	EnzPred-CPI	0.52 ± 0.02
	Ridge (Single-Task)	**0.58 ± 0.01**
Gylosyltransferase	ConPLex	0.52 ± 0.03
	EnzPred-CPI	0.46 ± 0.01
	Ridge (Single-Task)	**0.55** ± 0.00
Halogenase	ConPLex	**0.47 ± 0.04**
	EnzPred-CPI	0.38 ± 0.01
	Ridge (Single-Task)	0.44 ± 0.03
Phosphatase	ConPLex	0.33 ± 0.02
	EnzPred-CPI	0.39 ± 0.01
	Ridge (Single-Task)	**0.40 ± 0.00**

Table 5.8: Evaluation of different margin decay schemes on BindingDB.

Decay Scheme	Restart?	AUPR
No Contrastive Learning	N/A	0.664
Tanh	✓	**0.632**
Tanh	✗	0.609
Cosine	✓	0.620
Cosine	✗	0.602
Constant	✗	0.595

Chapter 6

Democratizing Protein Language Models: Parameter-Efficient Adaptation

6.1 Chapter Overview

While large machine learning models, such as the protein language models we discuss in this thesis, have had a transformative impact on computational modeling of proteins, the computation and memory requirements to train them remain a barrier to adoption for academic labs, biotech startups, and experimental labs with limited computational resources. Techniques for efficient, scalable training of such models would substantially increase their accessibility and adoption for a variety of proteomic tasks. In this chapter, we newly bring techniques for parameter-efficient fine-tuning from natural language processing to large protein language models, and demonstrate their efficacy as an alternative strategy to resource-intensive full fine-tuning for adapting to downstream tasks. We show that these methods remain competitive with significantly less computational cost and lay the groundwork for best practices for their use in proteomic modeling.

6.2 Introduction

The introduction of large pre-trained protein language models (PLMs) has transformed the computational modeling of protein sequence, structure, and function. These models are trained in an unsupervised manner on tens or hundreds of thousands of protein sequences, and they learn hidden representations which contain information about evolutionary constraints, chemical properties, secondary structure, and more [48]. These representations generalize broadly, which enables PLMs to be tuned to a wide variety of proteomic tasks. In the broader machine learning context, such large, pre-trained and task-agnostic models are referred to as "foundation" models. Typically, when a foundation model is tailored to a specific downstream task, the parameters of the pre-trained model are updated in a process known as fine-tuning, that adapts the parameters to fit the task-relevant supervised data (Figure 6.1a, top). However, as the size of foundation models increases, fine-tuning a model for a task of interest is increasingly computationally expensive, often with heavy GPU memory requirements. This puts such tuning out of the reach of many research groups, especially in academic, government, or startup environments. As increasingly large PLMs continue to be developed, such as the 6.4 billion parameter ProGen2 [246] or the 15 billion parameter ESM2 [147], the computational expense of fine-tuning will continue to be a pressing issue in proteomics.

Natural language processing (NLP) has seen a similar increase in foundation model size, with the largest NLP models approaching or exceeding one trillion parameters [247]. To address this hurdle, one recent NLP approach to fine-tuning (FT) uses prompt tuning [248], which circumvents updating the model's parameters altogether and instead updates additional input prompt embeddings, thus enabling the model to make "zero shot" predictions [249]; such an approach has been translated to generating protein sequences with some success [250]. Here, we draw inspiration from alternative methods developed for *parameter-efficient fine-tuning* (**PEFT**) in NLP (see [251] for a survey). These methods, usually focused on the transformer

layers of natural language models, add a small number (typically $< 1\%$ of total model size) of new parameters which are tuned, leaving the original model parameters untouched [252], [253]. These approaches require significantly fewer resources and can surprisingly sometimes achieve performance comparable to traditional fine-tuning (**FT**) [251] of all parameters in the layer in NLP tasks.

In this work, we introduce parameter-efficient fine-tuning methods to large protein language models, remarkably demonstrating performance competitive with or exceeding traditional fine-tuning using parameter-efficient training on two important proteomic tasks— homooligomer symmetry prediction and protein-protein interaction prediction. Homooligomers (proteins that form complexes with copies of themselves) often adopt symmetric conformations, and predicting the symmetry that a homooligomer adopts is an important task in structural biology and protein design [254] (Figure 6.1c). We newly show that as in NLP, a PEFT model slightly trails behind the performance of FT ($AUPR = 0.400$ vs. 0.489) but significantly outperforms baselines with frozen embeddings ($AUPR = 0.238$) and offers a much more compute-efficient alternative (Section 6.4.4).

We further explore how PEFT models perform in predicting general protein-protein interactions (PPIs) from primary sequence, an important and well-studied problem in proteomics [47], [255], [256] (Figure 6.1b). Here too, we find that PEFT models are competitive with FT models ($AUPR = 0.600$ vs 0.623). Intrigued by these findings, for both tasks we also tested the baseline model, which trains a multi-layer perceptron (**MLP**) classifier on embeddings from a frozen language model (Figure 6.1a, bottom). While MLP performance lags behind on symmetry prediction, we surprisingly find that this method actually outperforms both tuning methods ($AUPR = 0.684$) for PPI, demonstrating the continuing efficacy of simple downstream models in proteomics when PLM embeddings are used. In fact, we show that all three approaches for model adaptation outperform the current state-of-the-art on a gold-standard benchmark from Bernett et al. [257] on several metrics (Section 6.4.3).

We for the first time create a blueprint for applying PEFT methods to protein language

models on proteomics tasks by performing extensive experiments on LoRA hyperparameter choices (Figure 6.1d). Contrary to what is recommended for NLP tasks, we find that adding LoRA adapters to only the key and value matrices of the transformer achieves optimal performance (Section 6.4.6), and that performance drops off as the rank of adapter matrices drops below four (Section 6.4.6). Our work shows that it is possible to achieve competitive performance with significantly fewer resources than traditional approaches, opening up the power of protein language model fine-tuning to academic labs, small biotech startups and other research groups that lack substantial computational resources.

Language Modeling in Biology While the first protein language models (Bepler & Berger, UniRep) used recurrent neural networks like the bi-LSTM [48], [53], [258], recent work has converged around masked language modeling and the transformer. Models like ProtBert, ProtT5, [209], and ESM [208] are transformers trained on massive sets of protein sequence data in an unsupervised manner. These models learn meaningful representations which can be applied to replace manual feature engineering, or computationally expensive evolutionary searches and construction of multiple sequence alignments. Most recently, ESM2 [147] represents the largest protein language model to date, with models as large as 15 billion parameters. While this is still shy of the largest natural language models, this represents a significant step up in the size of protein language models and their capacity for unsupervised representation learning. While language modeling has seen the most success in proteomics, this success has seen language models expand to other aspects of biology. Biochemistry language models learn representations of small molecules [259], [260], most notably with ChemBERTa [261]. Likewise, the release of scGPT [262] has led to advancements in single-cell genomics, and language models have also seen direct clinical use, such as with the medical question answering model Med-PaLM [263].

Protein Structure One of the longest standing challenges in computational biology is that of protein structure prediction. While proteins are modeled by language models as

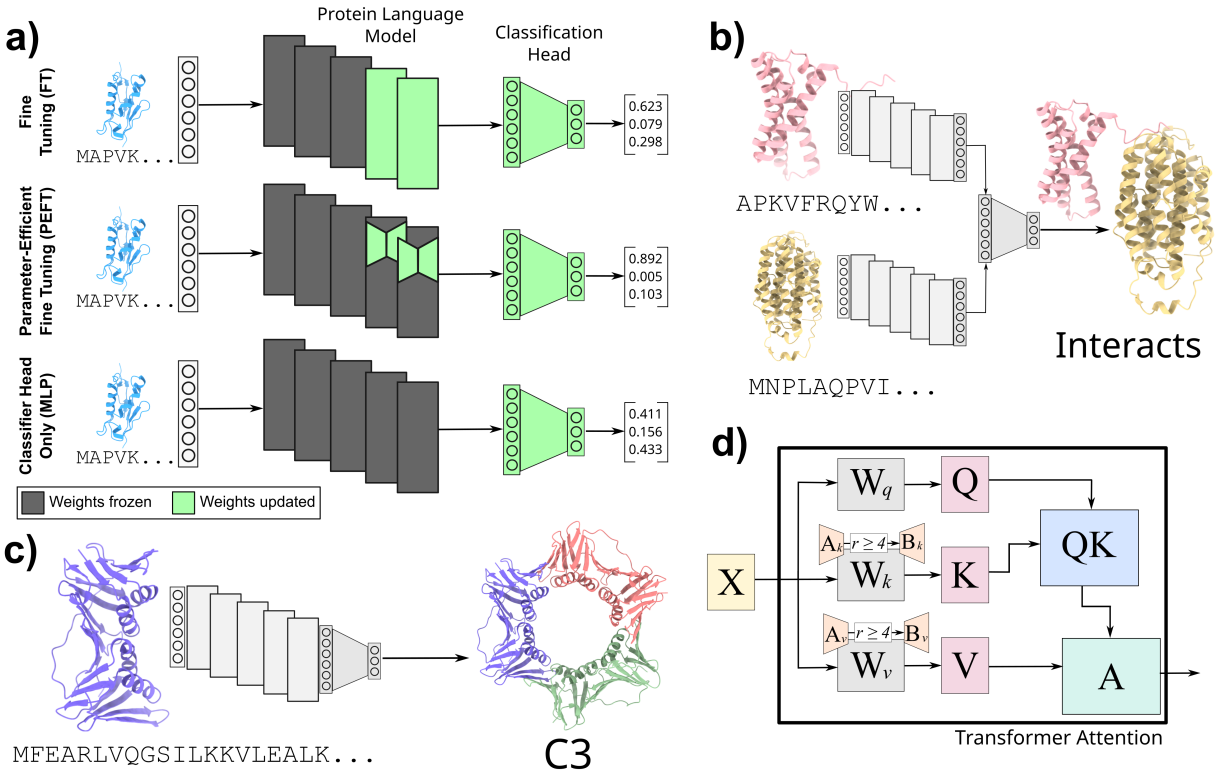


Figure 6.1: **Bringing PEFT to proteomics. (a)** The traditional paradigm for adapting protein language models to a specific downstream task is to fine-tune the parameters of the last n layers along with a new classification head (top, **FT**). Here, we introduce parameter-efficient fine-tuning (PEFT) to proteomics, tuning only the parameters of low-rank adapter matrices added to these final layers rather than the full weight matrices (middle, **PEFT**). We compare also with a baseline that uses the embeddings as-is and trains only the classification head (bottom, **MLP**) **(b)** We show that PEFT and MLP models achieve performance competitive with the state-of-the-art on predicting protein-protein interactions. **(c)** We also train PEFT models to predict the symmetry class of homooligomers, showing that the performance of PEFT models only slightly trails that of FT models yet uses several orders of magnitude fewer parameters. **(d)** We show the transformer self-attention with LoRA weight matrices added. We explore the hyperparameter space of low-rank adapters, creating a blueprint for best applying LoRA to protein language models. Notably, this blueprint differs from that in NLP. We recommend adding adapters with rank at least 4 to the key and value weight matrices of the self-attention layers. We explore different combinations of rank and the LoRA hyperparameter α in Section 6.4.6. X: input to attention head. $W_{q,k,v}$: query, key, and value weights. $A_{k,v}, B_{k,v}$: newly added low-rank adapter weights. Q, K, V: query, key, and value representations. QK: intermediate product of Q and K. A: output of attention layer.

sequences of amino acids (primary structure), they fold into secondary (alpha helices, beta sheets) and more complex tertiary structures, which imbues them with a variety of functions. For nearly 30 years the Critical Assessment of protein Structure Prediction (CASP) has

measured the ability to computationally predict the tertiary structure of a protein from its primary sequence. In 2020, AlphaFold2 [133], closely followed by RoseTTAFold in 2021 [148], presented a massive jump in performance, reaching near-experimental levels of accuracy. AlphaFold2 and RoseTTAFold use multiple sequence alignments (MSA) to incorporate evolutionary context into structure prediction, and recent methods like OmegaFold [146] and ESMFold [147] instead use pre-trained protein language models. While protein language model-based approaches have yet to reach the accuracy level of MSA-based approaches across the board, they nonetheless achieve extremely accurate performance and require only a single sequence. Because of this, they are much more computationally efficient, forgoing the need for the expensive MSA search step. In addition, these methods often outperform MSA-based methods on intrinsically disordered regions or proteins with little evolutionary context, and are competitive in terms of complex structure accuracy. Thus, protein language models represent an exciting step forward in tertiary structure modeling.

In addition to the structures that single chains fold into, proteins can form complexes known as quaternary structures comprising multiple protein chains. While methods like AlphaFold-Multimer [141] have attempted to fully model the structure of these complexes to moderate success, other methods have taken a different approach – focusing on the special case of homooligomeric proteins. In 2006, Levy et al. introduced 3DComplex [264], which categorizes protein complexes based on topology and symmetry. Recently, Schweke et al. introduced an atlas of protein homooligomerization [265], while QUEEN [266] attempts to predict the mutiplicity of such complexes.

Protein Interactions Cellular function is driven by a complex interplay of interactions between proteins. Experimental approaches to discern those interactions require substantial wet lab resources and time, which motivates the need for computational approaches to model protein interaction. Models such as AlphaFold-Multimer [141] have recently been developed to predict the structure of interacting complexes [151]. Quaternary structure prediction is

valuable if the pair is already known to interact, but often results in degenerate prediction for pairs which don't interact, and due to the size of the model is difficult to scale to the whole-genome and all possible protein pairs. Methods like PIPR [47], D-SCRIPT [255], Topsy-Turvy [12] and RAPPPID [256] predict protein-protein interaction (PPI) solely from widely-available primary sequence and are fast enough to run at genome scale. Recent work has begun to close the gap between whole-genome interaction prediction and complex structure modeling [13], [153], potentially unifying genome-scale PPI prediction with complex structure prediction. The Human Reference Interactome (HuRI) [167] remains the most complete experimentally-verified human protein interaction network.

6.3 Methods

6.3.1 Benchmark Data: PPI

While creating train/test splits based on filtering homologous proteins is common in machine learning for proteomics, the binary nature of PPI prediction presents a unique challenge because data leakage can still occur if only one protein of an interacting pair appears in both sets. If a so called "hub" protein with many interactions appears in both the training and test set, models can learn that this specific protein is likely to have positive interactions. Then, test set performance will be inflated even if nothing is learned about the actual pairwise interactions. After noting pervasive biases in previous benchmarks relating to sequence similarity and node degree, Bernett et al. [257] introduced a new gold standard data set for benchmarking PPI. The splits introduced in this benchmark apply a more stringent notion of sequence similarity for pairwise problems as introduced by Park and Marcotte [267], splitting by C3 similarity. In addition, both the positive and negative data sets are balanced with regard to node degree; as a consequence models cannot learn that proteins *in general* interact just because they are high degree. This data set consists of 163,192/59,246/52,035 training/validation/test edges, with an 1:1 ratio of positives to negatives.

6.3.2 Benchmark Data: Homooligomer Symmetry

The multiplicity and symmetry prediction task is formulated as a multi-class prediction problem. Given a protein chain, we classify it as one of 17 symmetry classes $C1$, $C2$, $C3$, $C4$, $C5$, $C6$, $C7-C9$, $C10-C17$, $D2$, $D3$, $D4$, $D5$, $D6-D12$, H, O, T, I (or "Unknown"). The C classes correspond to cyclic symmetries, the D classes to dihedral symmetries, and H, O, T, and I to helical, octahedral, tetrahedral, and icosahedral symmetries respectively. More extensive detail on different symmetry classes can be found within 3DComplex [264]. Protein sequences and structures were obtained from the Protein Data Bank (PDB) [268], as were their labels. Sequences were clustered at 30% sequence similarity and 80% coverage using MMseqs2 [159], and these clusters were used to define train, validation, and test splits. This method of splitting ensures that no two sequences that are highly similar will be in both a training and evalution set, which would allow the model to memorize sequence similarity rather than learning properties of sequence and structure that correspond with symmetry. This data consists of 370,986/46,833/102,978 training/validation/test edges. The support for each class in the text set is described in Table D.3.

We tracked performance of models through multiple metrics. It is difficult to pin performance to a single number with a multi-class classification, especially when the support for different classes is highly imbalanced. We compute the accuracy, F1 score, MCC, average precision (AUPR), precision, recall, and specifity for each class, as well as metrics averaged over each class. Note that because class support is highly variable (Section D.2.1) and resulting model performance varies widely, we use an unweighted ("macro") average to capture performance broadly across all classes.

6.3.3 Protein Language Model

We focused our efforts on ESM2, a transformer-based protein language model which is presently considered the state-of-the-art in protein language modeling [147]. ESM2 has

several different model sizes, ranging from eight million to 15 billion parameters. For this study, we focused on the 650 million parameter version of ESM2 (Section 6.4.2).

For an amino acid sequence $X = x_1x_2...x_n$, a PLM of dimension d returns a set of embeddings $E \in \mathbb{R}^{d \times n} = e_1e_2...e_n, e_i \in \mathbb{R}^d$. To standardize the size of representations for sequences of dynamic length, a pooling step needs to be undertaken. This is most commonly done either by averaging along the length of the sequences ($e_p \in \mathbb{R}^d = \frac{1}{n}\sum_{i=1}^{n} e_i$) or by selecting the first token of the sequence, a non-amino acid token (`[clsf]`) created specifically for sequence classification. Here, we chose to take the former approach as it explicitly integrates signal across the length of the protein. We note that while this is a commonly used approach, how to best aggregate sequence-length representations into a fixed dimension embedding is an open problem in language modeling (see NaderiAlizadeh and Singh for one recently proposed approach [269]). Converting this pooled embedding into a binary ($Y \in \mathbb{R}^2$) or multi-class ($Y \in \mathbb{R}^{18}$, 17 possible symmetry classes + "Unknown") prediction requires an additional classification head. For the PPI prediction task, fixed-length embeddings were averaged before being passed to the classification head, as in Szymborski et al. [256]. For the symmetry prediction task, there is only a single protein, so embeddings were passed directly to the classification head. In this study, we tested two different prediction heads. The first, applied directly to the pre-trained E without fine-tuning, is a simple multi-layer perceptron (MLP), with the number and size of layers determined by grid search (Section 6.3.5, Section D.1.4). The second is the `ESMClassificationHead` made available by the authors in the public HuggingFace repository, which consists of two dense layers with dropout and a tanh activation between the layers. We selected classification heads which were demonstrated to yield strong performance in previous work in order to minimize the need for hyperparameter search in this space.

6.3.4 Parameter-Efficient Adaptation

Houlsby et al. [253] introduced adapters, which add parameters in serial to each transformer layer, allowing for every layer of the model to be trained using only a small number of parameters. The current state-of-the-art is low-rank adapters (LoRA), introduced by Hu et al. [252], which adds two low-rank adapter matrices in parallel to the query and value weight matrices of the attention heads (Figure 6.1a, middle). LoRA adds two low-rank matrices A and B to each adapted weight matrix. Given weight matrix $W \in \mathbb{R}^{d\times k}$, LoRA adds new parameters $A \in \mathbb{R}^{r\times k}, B \in \mathbb{R}^{d\times r}, r \ll d, k$. The normal forward pass of the layer given input $x \in \mathbb{R}^k$ is $h = Wx$, and the forward pass with the LoRA adaptation is $h = Wx + BAx$. Only the weights of A, B are updated during back-propagation, while the weights of W are frozen. BAx is scaled by the quantity $\frac{\alpha}{r}$, where α is a hyperparameter which is held constant in the original report. While A is initialized with a random Gaussian distribution, by initializing $B = 0$ the first forward pass of the model is equivalent to the pre-trained model without adaptation. Following the recommendations of the original paper, we initially apply LoRA only to the query and value matrices of the attention head. We explore different combinations of weight matrix adaptation and different values of the rank r in Section 6.4.6. Even though α is originally held constant, we evaluate different settings of the α parameter as they relate to different rank values also in Section 6.4.6.

6.3.5 Training and Implementation

All PEFT and FT models were implemented in PyTorch (v.2.0.1), using the HuggingFace implementations of ESM2 from the `transformers` package (v.4.32.1) and LoRA from the `peft` package (v.0.5.0). Models were trained on NVIDIA V100 GPUs with 32GB of memory. We used the binary cross-entropy with logits loss to compute error, with an L2 weight decay of 0.01. Model weights were optimized via back-propagation using the Adam optimizer and a cosine decay with restarts learning rate schedule (initial learning rate 0.001 for MLP and

PEFT, 0.0005 for FT). Models were trained with an epoch size of $16,384$, with an on-device batch size of 4 and gradient accumulation every 16 steps, for an effective batch size of 64. PEFT and FT models were trained for 40 epochs and the best model based on validation AUPR was chosen for testing. Except for where otherwise specified, we used LoRA $r = 8$, LoRA $\alpha = 32$, LoRA dropout $p = 0.1$.

The hyperparameters for the MLP on language model embeddings with frozen weights were chosen by a grid search (implemented in scikit-learn (v.1.2.0), Section D.4). The best performing model had two hidden layers with sizes $(64, 64)$ and ReLU activations, and were optimized with the Adam optimizer for 2000 iterations with a tolerance of 0.0001 and an adaptive learning rate initialized at 0.01.

All model adaptation, training, and benchmarking code is made available open-source on GitHub at https://github.com/microsoft/peft_proteomics.

6.4 Results

6.4.1 Reduced memory usage of PEFT enables deeper fine-tuning

The most common approach to fine-tuning is to unfreeze the weights of the last n transformer layers, which we compare with adding LoRA adapter weights to the last n layers. Figure 6.2 shows the maximum GPU memory used by parameter-efficient fine-tuning (PEFT) or traditional fine-tuning (FT) for the last n layers of an ESM2 model trained on PPI prediction or homooligomer symmetry prediction (see D.2.2 for further discussion of the fine-tuning approach). Theoretically, tuning only intermediate layers without adjusting weights that rely on those intermediate representations reduces the degrees of freedom available to the model, and we show this to be empirically true as well (Section D.1.1). Both PEFT and FT eventually overflow available GPU memory as the number of layers tuned increases; tuning all layers of the model requires additional model parallelism with either approach. However, this parameter-efficient approach allows for training of deeper layers of the model

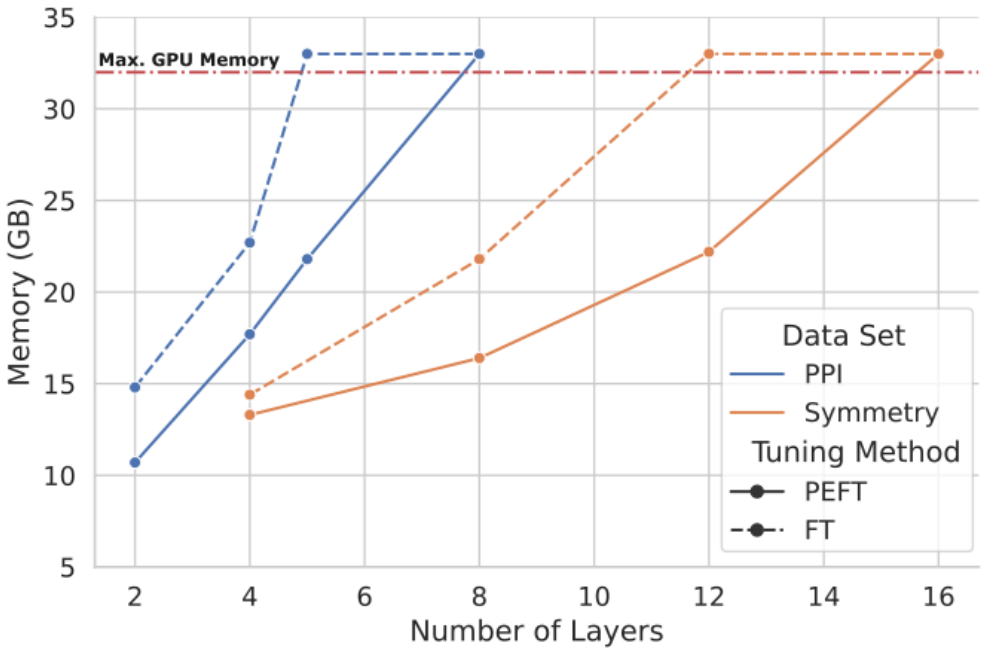


Figure 6.2: **PEFT training requires reduced GPU memory.** We compare the maximum GPU memory usage of parameter-efficient fine-tuning vs. traditional fine-tuning different numbers of transformer layers. All values are reported in GB. Values above 32 GB (the red, dash-dot line) indicates that the run was killed due to running out of GPU memory. More layers are able to be adapted across the board for the symmetry prediction task since it requires only a single protein, rather than a pair. For the same number of layers, PEFT models require less GPU memory, thus parameter-efficient fine-tuning enables adaptation of deeper model layers within the same limits of computational resources. We show in Section D.1.2 that this deeper training often yields improved performance.

while staying within a lower memory budget. Because the PPI task requires storing the compute graph for a pair of proteins, the marginal impact of PEFT methods on memory consumption is less than for the symmetry task, where using LoRA to adapt weight matrices allows for several additional layers to be trained. All experiments were performed on a GPU with 32GB of memory, with a batch size of 4 and maximum sequence length of 1024. In general, we show that adaptation of deeper layers results in a corresponding increase in performance. We specifically adapt increasingly many layers from the end of the model, rather than intermediate layers (Section D.1.2).

6.4.2 650M vs. 3B ESM2 model

Typically in language modeling, larger models yield better performance leading to training or increasingly large models. We trained a multi-layer perceptron classifier (MLP) on embeddings from both 650M and 3B parameter models with frozen weights to predict protein-protein interactions. We found that despite having 4.5x fewer parameters, the 650M parameter model actually performs slightly better (Table 6.1). This indicates that even with reduced compute capacity available, smaller foundation models may be sufficient to achieve good performance on proteomics tasks—and that the limiting factor for performance may not simply be the scale of models. Consequently, all results presented elsewhere in this manuscript use the 650M parameter version of ESM2. We discuss the question of foundation model size in Section 6.5.

Table 6.1: Test set performance of a MLP classifier trained on pooled embeddings from the 650 million and 3 billion parameter versions of ESM2 with frozen weights. While the two models are competitive, the 650M parameter version outperforms the larger 3B parameter version in accuracy, MCC, AUPR, precision, and specificity. The 3B parameter model achieves a higher F1 score and recall.

	Accuracy	**F1**	**MCC**	**AUPR**	**Prec.**	**Rec.**	**Spec.**
650M	**0.631**	0.632	**0.261**	**0.684**	**0.630**	0.633	**0.623**
3B	0.607	**0.650**	0.221	0.656	0.586	**0.730**	0.484

6.4.3 Efficient classifiers achieve state-of-the-art PPI prediction

We use parameter-efficient fine-tuning (**PEFT**) to train an ESM2 model to predict protein-protein interactions from sequence on the benchmark data set from Bernett et al. [257]. We compare with a model that is fine-tuned in the traditional way (**FT**), and with a baseline that trains a classifier on sequence embeddings from the ESM2 model with frozen weights (**MLP**). See Figure 6.1a for an overview of these three approaches. In addition, we compare to the best prior scores from Bernett et al. [257]—either Topsy-Turvy [12], or SVM-PCA, a baseline constructed by Bernett et al. [257] which trains a support vector machine on

PCA-reduced sequence similarity vectors. Note that for each benchmark metric, we selected the best score across *all* methods evaluated, and that no single method achieved the "Best Prior" performance across the board, so this is a significantly higher threshold than comparing to any single method.

Table 6.2 shows the performance of these models. Both the PEFT and MLP models achieve performance competitive with the FT model, despite having several orders of magnitude fewer parameters. In fact, the MLP model achieves the best overall accuracy, MCC, and AUPR on these benchmarks, while the PEFT model achieves the best F1 score and recall. Surprisingly, both the MLP and PEFT models outperform the "Best Prior" methods in these metrics. These results suggest that parameter-efficient methods provide competitive alternatives and can unlock the predictive power of protein language models—and provide additional evidence that simple scaling of model size is not sufficient for proteomic tasks [270] (see also Section 6.4.2). Both PEFT and MLP models use less GPU memory than the FT model (MLP substantially so) and the MLP model requires significantly less training time.

6.4.4 Predicting homooligomer symmetry with PEFT

We additionally train PEFT, FT, and MLP models to predict homooligomer symmetry. While this task also involves learning representations that capture protein structure, it is fundamentally different from the PPI task because it requires learning on only a single protein, rather than a pair. As we show in Table 6.3, both the PEFT and FT models significantly outperform the MLP model on all metrics. Traditional fine-tuning still yields the best performance, but parameter-efficient fine-tuning using LoRA is a viable alternative—performance is within 10-15% of the FT model, and the model uses 3 orders of magnitude fewer parameters. The PEFT model actually achieves the best MCC and specificity of the three.

Unlike the binary prediction task of PPI prediction, homooligomer symmetry prediction is a multi-class problem, where we consider 17 different possible symmetries (and an 18th

Table 6.2: **Applying PEFT to train models for protein-protein interaction.** We trained multiple variants of ESM2 to predict protein-protein interactions, and evaluate using the benchmark data sets from Bernett et al. [257]. **MLP** indicates a multi-layer perceptron trained on embeddings from a frozen model, while **PEFT** and **FT** indicate parameter-efficient fine-tuning and traditional fine-tuning of the transformer layers. Due to the reduced memory footprint of PEFT, we were able to fine-tune an additional layer. Simply using ESM2 embeddings with an MLP classifier outperforms the best prior methods across most metrics. The PEFT model achieves increased recall and F1 score compared to the MLP model, and also out-performs the best prior. Both MLP and PEFT are competitive with fine-tuning.

	Best Prior	**MLP**	**PEFT** (5 Layers)	**FT** (4 Layers)
# Trainable Params.	-	88,769	368,897	78,873,857
Validation				
Accuracy	-	0.595	0.596	**0.597**
F1	-	0.576	**0.648**	0.612
MCC	-	-	**0.201**	0.194
AUPR	-	**0.632**	0.620	0.622
Precision	-	**0.603**	0.574	0.590
Recall	-	0.552	**0.742**	0.636
Specificity	-	-	0.450	**0.557**
Test				
Accuracy	0.56 (Topsy-Turvy)	**0.631**	0.608	0.604
F1	0.61 (SVM-PCA)	0.632	**0.666**	0.631
MCC	0.15 (Topsy-Turvy)	**0.261**	0.230	0.210
AUPR	-	**0.684**	0.600	0.623
Precision	**0.65** (Topsy-Turvy)	0.630	0.580	0.591
Recall	0.77 (SVM-PCA)	0.633	**0.780**	0.676
Specificity	**0.86** (Topsy-Turvy)	0.623	0.436	0.532

"Unknown" class). This makes it more difficult to summarize model performance into a single number—especially because the number of examples of different classes in the test set is so variable (Table D.3). The numbers reported in Table 6.3 are macro averages across all classes, but in Figure 6.3 we show the accuracy, AUPR, and F1 score of the three models broken down by true class. These results make it clear that the improved performance of the PEFT and FT models relative to the MLP baseline comes primarily from their increased ability to learn about rare classes. On common classes like C1, C5, D2, and I, the MLP model

Table 6.3: **Applying PEFT to train models for homooligomer symmetry.** We trained multiple variants of ESM2 to predict the symmetry of homooligomers from primary sequence. **MLP** indicates a multi-layer perceptron trained on embeddings from a frozen model, while **PEFT** and **FT** indicate parameter-efficient fine-tuning and traditional fine-tuning of the transformer layers. Due to the reduced memory footprint of PEFT, we were able to fine-tune four additional layers (12 vs. 8). While traditional fine-tuning still yields the best performance, PEFT is competitive across many metrics and significantly outperforms training only the classification head.

	MLP	**PEFT** (12 Layers)	**FT** (8 Layers)
# Trainable Params.	89,857	657,810	157,585,810
Accuracy	0.206	0.393	**0.413**
F1	0.228	0.405	**0.455**
MCC	0.299	**0.464**	0.460
AUPR	0.238	0.400	**0.489**
Precision	0.341	0.477	**0.618**
Recall	0.206	0.393	**0.413**
Specificity	0.960	**0.970**	**0.970**

achieves accuracy competitive with the PEFT model and often even the fine-tuned model. However, the larger complexity of the FT model allows it to also learn about classes like C3, C4, D6-D12, and O significantly better than the less complex models. This may contribute to explaining wy performance between the three models is much closer in the balanced PPI prediction task.

6.4.5 Visualizing attentions after fine-tuning

In Figure 6.4, we show the impact of parameter-efficient fine-tuning on attention (A matrix from Figure 6.1d). Here, we look at the PEFT model trained for PPI prediction from Table 6.2. We visualize attention from the five LoRA-adapted layers averaged across all heads. Figure 6.4a shows attention from the pre-trained model for NADH dehydrogenase 1 β subcomplex subunit 1 (UniProt ID: O75438) (Figure 6.4c). Attention is concentrated along the diagonal. In contrast, after PEFT with LoRA weights the attention is much more diffuse across the length of the protein, especially in the later transformer layers (Figure

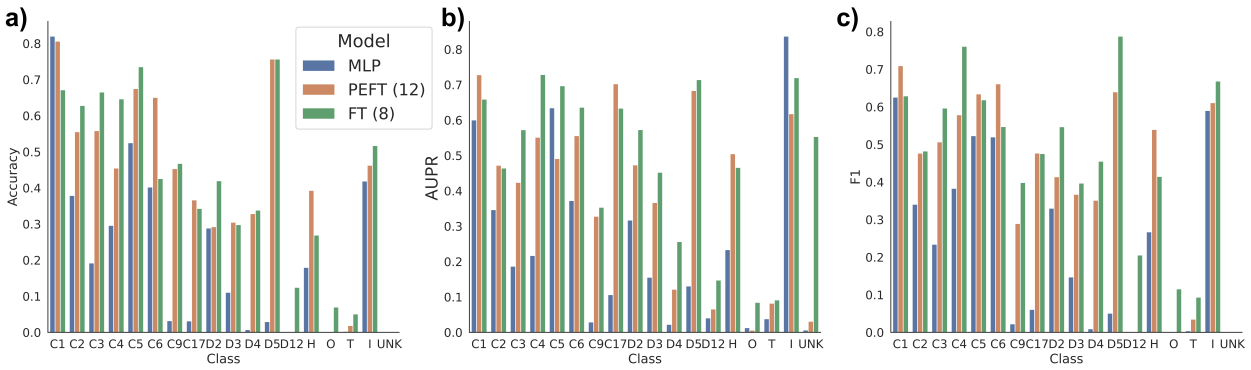


Figure 6.3: **Breakdown of model performance by homooligomer symmetry class.** Looking at only a single set of metrics can be helpful to get an aggregate idea of model performance, but hides complexity in the case of a multi-class classification task such as homooligomer symmetry prediction. Here, we show the per-class accuracy **(a)**, AUPR **(b)**, and F1 **(c)** of a classifier trained with frozen ESM2 embeddings (**MLP**), a model fine-tuned with LoRA (**PEFT**, 12 layers), and a model where all paramters in the final 8 layers were fine-tuned (**FT**). **MLP** performance is unsurprisingly relatively strong for high-support "easy" classes (e.g. C1, C5, D2, I), but substantially worse for rarer classes (e.g. C7-9, C10-C17, D4, D5). The **PEFT** model is competitive for most classes, but generally lags behind the performance of the **FT** model.

6.4b). This suggests that for PPI prediction, the LoRA weights allow for more distant amino acids to attend to one another. We show one representative example here, but this diffusion of attention occurs broadly; we show another representative example (NADH dehydrogenase 1 β subcomplex subunit 10, UniProt ID: O96000, which interacts with O075438 in our test set) in Figure 6.5.

We can quantify this effect by computing the sample Pearson correlation of these attention values. In summary, this value treats attention values at amino acid positions i, j as samples from distributions X, Y and computes the correlation of these samples (see Section D.3 for details). This value measures the extent to which attention is concentration near the diagonal, capturing local vs. global effects of attention. We show a decrease in correlation along the diagonal after fine-tuning (Figure 6.4d) for the five layers which are adapted using LoRA. We then compute the this correlation for the pre-trained PLM and the adapted PLM for all 3,020 proteins. Figure 6.4e shows the distribution of differences in scores for each layer, where a higher value indicates more global attention. For all layers adapted, there

is a significant difference in the diagonal concentration of attention after the application of parameter-efficient fine-tuning. This suggests that the representations produced by the language model are able to predict PPI in part by increasing the relative impact of long-range attention. This effect is similar to what is seen when fine-tuning all parameters of the PLM (Figure D.5). We show an additional representative example in Figure 6.5, the interacting partner NADH dehydrogenase 1 β subcomplex subunit 1. This effect is less noticeable in PEFT models trained to predict homooligomer symmetry, where there is a slight diffusion of attention but it still remains largely concentrated along the main diagonal (Figure 6.6).

6.4.6 Impact of LoRA hyperparameters on performance

Weight matrix selection In their original manuscript on LoRA, Hu et al. [252] show that for natural language models, adding low-rank adapter matrices to only the query and value weights (W_Q, W_V) of the attention heads yields the best tradeoff of performance and parameter-efficiency. However, the space of natural language is not necessarily the same as that of protein sequence, and we sought to evaluate to what extent the choice in adapted weight matrices affects performance. Table 6.4 shows that while performance is relatively robust to the choice of weight matrices, adapting the key (W_K) and value (W_V) matrices results in the best overall performance. We note that the value matrix alone achieves similar results while also being a more parameter-efficient. Thus, that could be the best choice for parameter-efficient fine-tuning of PLMs, if memory constraints are especially tight. When applying LoRA to protein language models, we recommend adding adapter matrices to the key and value weight matrices.

LoRA matrix rank selection The rank r of the newly added LoRA weight matrices plays an important role in the performance of PEFT models and the memory that they require. Hu et al. [252] show that LoRA remains effective at extremely low ranks, with competitive performance even when $r = 1$. However, the effectiveness of different rank values is dependent

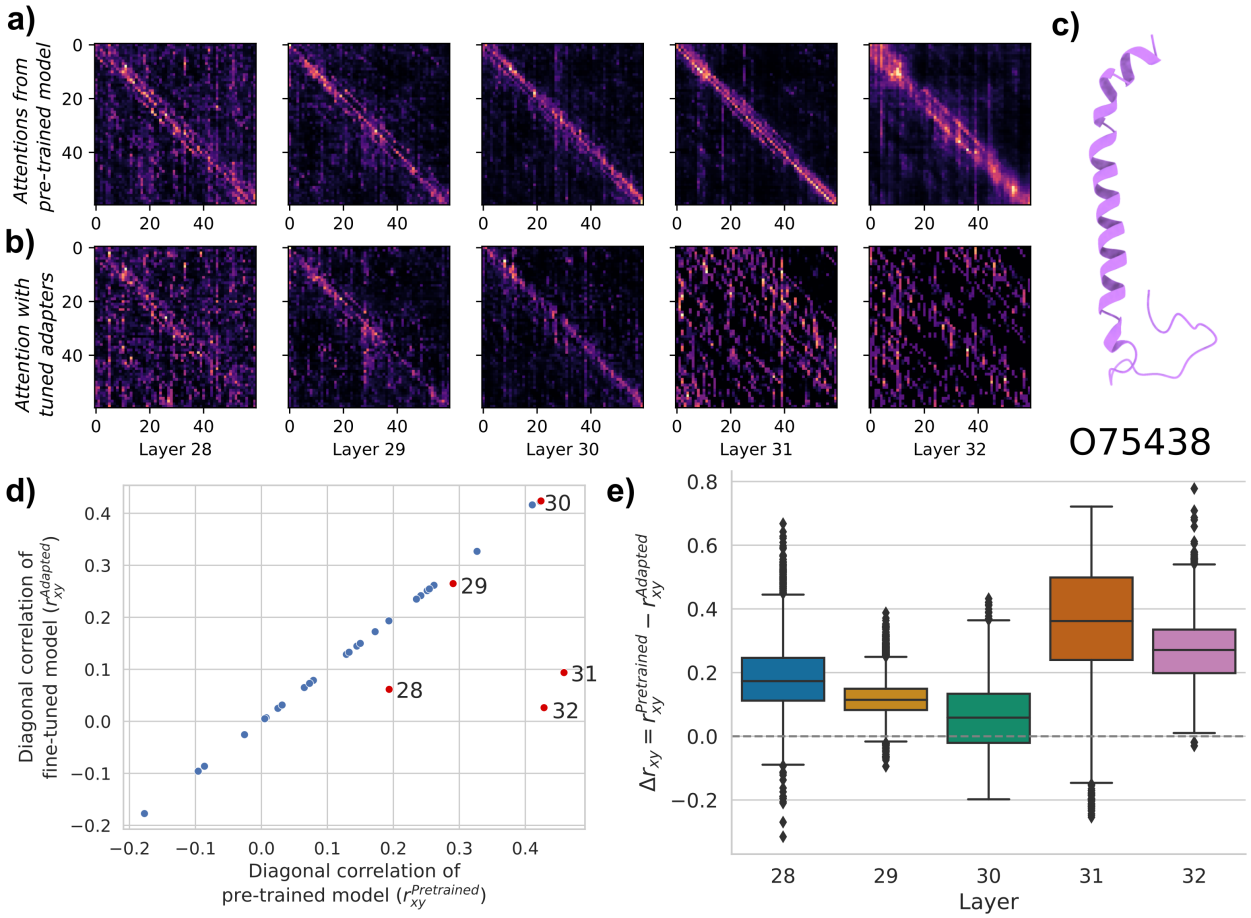


Figure 6.4: **Visualizing attention matrices. (a)** Attentions for NADH dehydrogenase 1 β subcomplex subunit 1 (UniProt ID: O75438) using the pre-trained ESM2. **(b)** Attentions for the same protein after parameter-efficient fine-tuning. **(c)** Structure of O75438. We find that PEFT weights result in attention that is more spread out across the length of the protein when trained for PPI prediction. **(d)** For each of the attentions, we compute the extent to which attention is concentrated along the diagonal (local) as opposed to diffused (global) using the weighted sample correlation (r_{xy}). For the five layers which are adapted with PEFT, this diagonal correlation is significantly lower indicating more globally diffuse attention. **(e)** We compute the difference between the diagonal correlation of the base and adapted attentions (δr_{xy}) for all 3,020 proteins in our PPI data set. We find that attentions in every layer are significantly less diagonal after fine-tuning, with the strongest impact in the final two layers.

on both the intrinsic dimension of the language and the task [271], and robustness in rank variance will not necessarily hold for proteomic sequences and inference tasks. In Table 6.5, we show the results of training PEFT models with LoRA rank $r = 1, 2, 4, 8, 64$ (following Hu et al. [252]) to predict protein-protein interactions. We find that the best model performance

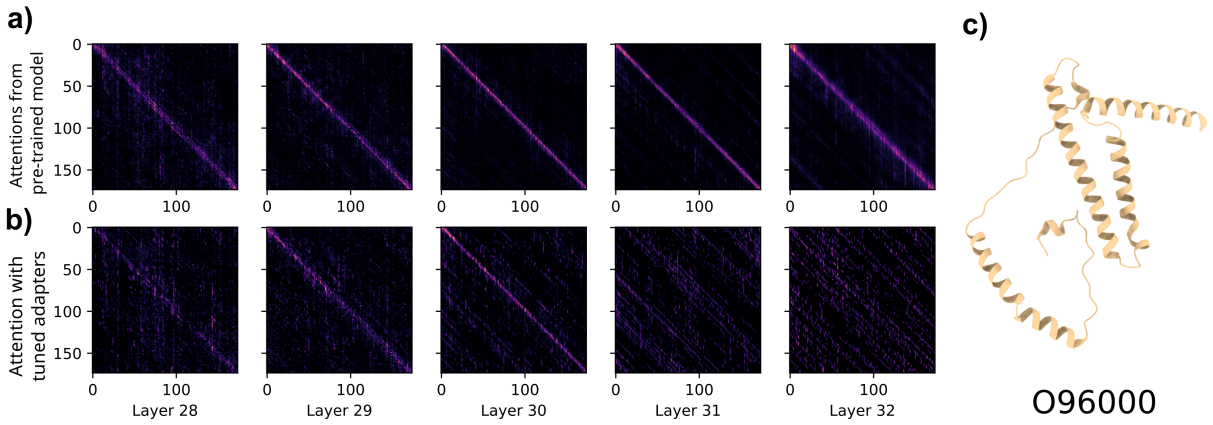


Figure 6.5: **Visualizing attention matrices.** **(a)** Attentions for NADH dehydrogenase 1 β subcomplex subunit 10 (UniProt ID: O96000) using the pre-trained ESM2. **(b)** Attentions for the same protein after parameter-efficient fine-tuning. **(c)** Structure of O96000. We find that PEFT weights result in attention which is more spread out across the length of the protein when trained for PPI prediction.

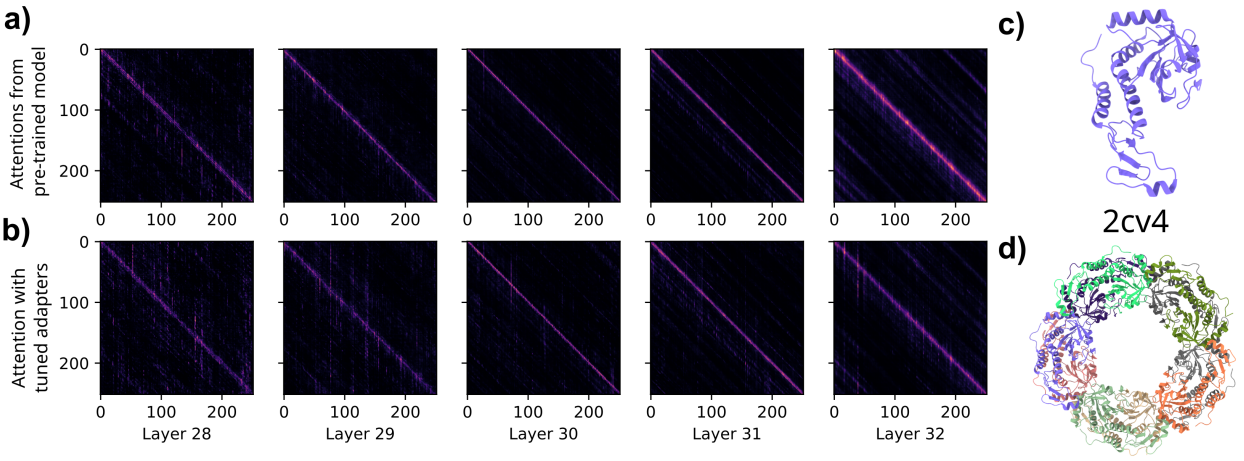


Figure 6.6: **Visualizing attention matrices.** **(a)** Attentions for an Archael Peroxiredoxin from the Aerobic Hyperthermophilic Crenarchaeon Aeropyrum pernix K1 (PDB ID: 2CV4), which adopts a dihedral D5 symmetry, using the pre-trained ESM2. **(b)** Attentions for the same protein after parameter-efficient fine-tuning. **(c)** Structure of a single subunit 2CV4. **(d)** Structure of the 2CV4 homooligomer, with D5 symmetry. Here, find that PEFT weights result in attention which is only slight spread more spread across the length of the protein, but still remains concentrated near the diagonal.

is achieved at $r = 4$, but that higher ranks yield similar performance. For ranks less than 4, model performance is still strong, but is noticeably reduced. We perform the same experiments on homooligomer symmetry prediction, with similar findings—although $r = 8$ is better for this task. We show in Table 6.6 the result of training models with different LoRA rank values for homooligomer symmetry prediction. Here, $r = 8$ is the best performing, closely followed by $r = 4$. Performance drops off noticeably with $r < 4$.

LoRA α selection In their original report, Hu et al. [252] state that the hyperparameter α functions similarly to a learning rate, and that they set it to the first rank value tried and hold it constant. However, we sought to evaluate whether tuning this parameter α could have an impact on performance. In Table 6.7, we show various combinations of α and rank applied to the symmetry prediction task.

6.5 Chapter Perspectives

In this work, we bring parameter-efficient fine-tuning methods to protein language models. We show that PEFT models achieve performance comparable to traditional fine-tuning, while requiring a reduced memory footprint, which enables training deeper layers of the model. The competitive performance of PEFT models holds both for PPI prediction, a balanced, binary task, as well as symmetry prediction, a multi-class and highly unbalanced task. Our work democratizes protein language models by charting a path for fine-tuning these models using substantially fewer parameters and less GPU memory. As the scale of PLMs continue to increase, it will become increasingly important to use these and other approaches when traditional fine-tuning is computationally infeasible.

Additionally, while approaches from natural language processing have so far transferred quite well, the distributions and intrinsic dimensions of protein language are clearly different than natural language. We show one consequence of this; our recommendation for LoRA rank, α, and which attention matrices to adapt differs from the original conclusions of Hu

et al. [252]. Notably, LoRA when applied to proteomics does not seem to admit as low of a rank as suggested in the initial report. In considering the question of which matrices to adapt, while we see consistent results between the two tasks we test, it is possible that these conclusions will vary with additional tasks. While our recommended parameters offer a strong starting point, it remains important to test these parameters with a validation set to achieve maximum performance when adapting large foundation models. Further, while LoRA is currently the most widely-used PEFT method for natural language, other PEFT methods exist which may better suit the space of protein language such as the adapter method of Houlsby et al. [253]. QLoRA [272] is an especially promising approach that performs model quantization to substantially reduce memory usage. This quantization will likely be especially valuable for natively memory-intensive tasks, such as modeling extremely long proteins or prediction of protein complexes. It may be necessary in the future to develop parameter-efficient approaches specific to protein language.

The successful transfer for PEFT from large language models to PLMs suggests that other efficiency techniques such as quantization may yield similar performance gains [273], [274]. While this work has focused on adaptation of protein language models, large protein structure models such as OmegaFold, RosettaFold, ESMFold, and AlphaFold [133], [146]–[148] have begun to see use as foundation models and could be similarly amenable to tuning for downstream tasks with PEFT methods. In addition, it remains to be shown whether PEFT methods are equally competitive for language models trained on other types of biological sequences, such as DNA [275] or SMILES strings [261].

This work also shows the limitations of scale in protein language modeling. Not only are PEFT and MLP models able to achieve state-of-the-art performance on PPI prediction, but we show that embeddings from the 650 million parameter version of ESM2 outperform those from the 3 billion parameter version (Section 6.4.2). While larger models certainly enable better performance, they do not guarantee it; we emphasize the continued need for models which can be easily trained and run by even small research groups, including primarily

biological groups with constrained computational resources. These limitations of scaling laws for protein language models also suggests that data, rather than compute, is presently the bottleneck in performance. The training data sets used for PLMs are still orders of magnitude smaller than the largest natural language training data sets. Thus, an increase in the number and diversity of protein sequences, such as from metagenomes [147], [276], could drive improved performance for larger PLMs. The inference-time efficiency of the MLP method, which only requires a single forward pass through the language model per protein, is especially valuable for tasks which require ultra-massive prediction, such as large compound library screens [14], [277]. Especially with the cost and environmental impact of large-scale language model training and tuning [278], parameter-efficient tuning or methods which use frozen embeddings should remain a viable alternative for tailoring foundation models to a specific task, and should be tested alongside traditional fine-tuning approaches.

Table 6.4: **Adaptation of different sets of transformer weights.** Hu et al. [252] recommend adding low-rank adapters to the query and value matrices of the attention heads for natural language. We investigate whether the same recommendation holds for the space of protein language, specifically on the task of protein-protein interaction prediction. We report the validation AUPR, which was used to select the best epoch for test set evaluation, and the AUPR, accuracy, F1 score, MCC, precision, recall, and specificity on the test set for all different combinations of the query (W_Q), key (W_K), and value (W_V) matrices. We find that adapting the W_K and W_V together yields the best performance. Adapting only the value matrix also performs well and uses fewer parameters.

Weight Matrix	# Trainable Params.	Val. AUPR	AUPR	Acc.	F1	MCC	Prec.	Rec.	Spec.
W_Q	266,597	0.536	0.529	0.516	0.430	0.034	0.524	0.364	0.669
W_K	266,597	0.565	0.562	0.544	0.433	0.095	0.572	0.348	**0.739**
W_V	266,597	0.612	0.610	**0.605**	**0.650**	**0.216**	0.583	**0.735**	0.474
W_Q, W_K	368,897	0.590	0.576	0.564	0.582	0.129	0.559	0.607	0.521
W_Q, W_V	368,897	0.619	0.617	0.599	0.633	0.201	0.583	0.692	0.506
W_K, W_V	368,897	**0.637**	**0.633**	0.601	0.630	0.205	**0.587**	0.680	0.522
W_Q, W_K, W_V	471,297	0.628	0.613	0.603	0.639	0.210	0.585	0.704	0.502

Table 6.5: **Variation in rank is tolerated, but performance degrades at too low a rank.** Hu et al. [252] show that LoRA is robust to rank r values as low as one. We test whether this robustness holds for protein language and on the PPI prediction task. Test set performance remains relatively strong across all ranks tested and actually peaks at $r = 4$, but there is a noticeable drop-off in performance for $r = 1, r = 2$. We recommend using a rank of at least 4 when fine-tuning protein language models.

Rank	**# Trainable Params.**	**Val. AUPR**	**AUPR**	**Acc.**	**F1**	**MCC**	**Prec.**	**Rec.**	**Spec.**
1	189,697	0.621	0.600	0.592	0.635	0.190	0.575	0.710	0.474
2	215,297	0.619	0.586	0.595	0.614	0.190	0.586	0.644	**0.545**
4	266,497	**0.644**	**0.633**	**0.604**	**0.658**	**0.219**	0.579	0.761	0.447
8	368,897	0.637	**0.633**	0.601	0.630	0.205	0.587	**0.680**	0.522
64	1,802,497	0.638	0.623	0.601	0.630	0.205	**0.588**	0.679	0.523

Table 6.6: **Robustness in rank also holds for homooligomer symmetry prediction.** We perform the same hyperparameter search over rank as in Section 6.4.6, this time training models to predict homooligomer symmetry. We find that like for PPI prediction, while performance is respectable at all values, it drops off noticeably for $r < 4$. For symmetry, rank $r = 8$ is the best performing, and there is actually a slight drop off with rank $r = 64$.

Rank	**Val. AUPR**	**AUPR**	**Acc.**	**F1**	**MCC**	**Prec.**	**Rec.**	**Spec.**
1	0.506	0.359	0.345	0.352	0.428	0.419	0.345	0.968
2	0.525	0.385	0.351	0.369	0.450	0.516	0.351	0.969
4	**0.531**	0.403	0.372	0.383	0.455	0.503	0.372	0.969
8	0.461	**0.416**	**0.390**	**0.430**	**0.468**	**0.558**	**0.390**	**0.970**
64	0.445	0.388	0.359	0.358	0.420	0.444	0.359	0.968

Table 6.7: We report here the test set AUPR of several model trained with PEFT on the symmetry prediction task, with varying values of the α and rank hyperparameters of LoRA. While performance does vary with α and rank, there is not a clear correlation between the two.

α**Rank**	**1**	**2**	**4**	**8**	**64**
16	0.395	0.404	0.394	0.399	0.377
32	0.395	0.385	0.403	0.416	0.387
64	0.414	0.429	0.420	0.423	0.430
100	0.376	**0.436**	0.422	0.431	0.418

Chapter 7

Discussion

We have included at the end of each chapter a short section offering perspectives on that chapter—the significance and limitations of the work, and opportunities for future advancements towards the specific problem. Here, we conclude this thesis with a summary of this thesis as a whole, and with broader thoughts on exciting directions in computational modeling of biological systems and the state of the field as a whole.

7.1 Thesis summary

In this thesis, we have shown how computational advances in the modeling of language can be adapted to contribute towards the understanding of molecular interactions. While the history of computational models of biological sequence is long and rich (see Section 2.3), protein language models are an exciting new frontier due to their ability to integrate long-range contextual information, their capacity to broadly represent sequence space, and their use as highly informative protein feature generators for downstream models through vector embeddings. In this work we advance this technology beyond modeling of single proteins to the pairwise problem of interactions, contributing substantially to computational systems biology. We make new contributions towards the computational prediction of protein-protein interaction networks in Chapters 3 and 4, and computational prediction of protein-small

molecule interactions in Chapter 5. In Chapter 6, we begin the work of increasing the accessibility of these models (see also Section 7.3.2) by introducing methods for parameter-efficient fine-tuning to protein language models. Taken as a whole, these chapters demonstrate the promise of unsupervised modeling on protein sequences for making sense of evolution, of the diversity of sequence space, and of the sequence-structure-function relationship as they pertain to the biochemistry of interaction and binding.

7.2 Extensions of this work

There remain several open problems in computational systems biology, especially where prediction of networks overlaps with advances in structural biology. The work described in this thesis has primarily focused on sequence modeling, although we describe integrative approaches which incorporate network level information, as well as explicit information on protein structure (Chapter 4). The future of computational interaction modeling is **multimodal**—and there remains a need for models which can effectively synthesize multiple different angles and discrete sources of information about problems. Just as advances in joint modeling of text and images has led to revolutionary advances such as CLIP [279] and DALL-E [280], recent work in joint modeling of sequence and structure [281]–[285] represent an exciting opportunity to improve interaction prediction, both for proteins and small molecules.

The future of interaction prediction is also **contextual**. We have discussed at length in this thesis the sequence-structure-function relationship for proteins, but this is a simplification of the true complexity of interaction. Several factors determine the structure, and thus the interactions for a protein beyond its sequence. The environmental temperature [286], [287], pH [288], [289], and any post-translational modifications [290], [291] will necessarily change the energy landscape that governs protein structure. Methods which consider these external factors [292] will play an important role in accurate biophysical modeling. Even given two

structures, the affinity of their interaction (which remains challenging to predict, see Section 5.4.8) will also depend on these external factors, as well as the presence of enzyme co-factors or any competitive binders. Thus, integration with genetic and cell-type specific transcriptomic information [262] will be necessary to fully capture the context of an interaction.

Finally, the future of interaction prediction is **dynamic**. Because proteins are flexible, modeling them with a single structure is a simplification of the biochemistry at work during interactions. Advances in experimental technologies such as cryo-EM [293] and single-molecule FRET [294], [295] have allowed us to gain a deeper understanding of the set of conformations that a protein can exist in, as well as the dynamics of these transitions. Computational advances in modeling of protein ensembles [296]–[298] are an exciting frontier in structural biology, and their integration into predictions of protein binding and interaction will likewise lead to more accurate models of biological systems.

7.3 The promise and practice of computation in biology

Over the past three decades, computational algorithms and models have gone from a convenient option for analyzing biological data to an essential part of biological discovery. In this section, we offer some thoughts on the effective synthesis of the two fields, with a particular focus on the techniques introduced in this thesis.

7.3.1 On computation as hypothesis generation

We discuss in the introduction the value of efficient models which are able to scale to the complexity of the genome (Section 1.2). However, high-throughput and efficient models also will play a crucial part in hypothesis generation. The canonical experimental loop in science consists of proposing a hypothesis, then testing in an attempt to falsify it. However, as the scale of biological data grows increasingly large and complex it can be difficult for humans to wrangle this complexity, and to know which tests are likely to succeed. Computational

models are able to find patterns in this data, and propose tests to run—the data from which is then often fed back into the model, in what is known as "lab-in-the-loop" computation [299]. Critical to this loop is the ability to quickly sift through candidate hypotheses to identify the most promising, and then tightening the feedback loop with experimental results. Scalable models such as those presented in this thesis—predicting interactions which are likely to occur, suggesting clusters of co-functional proteins, and identifying novel candidate therapeutics—enable the former and can be readily tested at the bench. Methods which are able to model uncertainty [122] are likewise important for determining which tests to run to most efficiently increase the quality of model predictions.

7.3.2 On the accessibility and interpretability of models

A tool is only as useful as it is accessible. Because the majority of computational developers are not themselves biological practitioners, there exists an accessibility gap where many "state-of-the-art" methods go unused by those involved in biological discovery—due either to a lack of computational resources (see Chapter 6) or knowledge. Efforts to make computational advances widely and easily available, such as ColabFold [300] are an essential and often under-met need. Tools such as Google Colab [301] and Gradio [161] make it allow computational practitioners to make their tools available in a web-browser with relatively little overhead, and, in addition to open-source code sharing on repositories such as GitHub, these should be considered an essential part of the computational biology toolbox. In addition, interpretable or "white-box" models and advances in explainable AI [302] are especially important in biology, where the goal is often not just prediction but understanding of the underlying system. The work in this thesis attempts, where possible, to make model outputs explainable, and to make trained models easy to run for those with limited computational resources or experience.

References

[1] M. Neveu, H.-J. Kim, and S. A. Benner, "The "strong" rna world hypothesis: Fifty years old," *Astrobiology*, vol. 13, no. 4, pp. 391–403, 2013.

[2] N. Kozlyuk, A. J. Monteith, V. Garcia, S. M. Damo, E. P. Skaar, and W. J. Chazin, "S100 proteins in the innate immune response to pathogens," in *Calcium-Binding Proteins of the EF-Hand Superfamily*, Springer, 2019, pp. 275–290.

[3] M. Schneider, A. G. Zimmermann, R. A. Roberts, *et al.*, "The innate immune sensor NLRC3 attenuates Toll-like receptor signaling via modification of the signaling adaptor TRAF6 and transcription factor NF-κB," *Nature Immunology*, vol. 13, no. 9, pp. 823–831, 2012. DOI: 10.1038/ni.2378.

[4] T. C. Bruice and S. J. Benkovic, "Chemical Basis for Enzyme Catalysis," *Biochemistry*, vol. 39, no. 21, pp. 6267–6274, 2000. DOI: 10.1021/bi0003689.

[5] L. M. Veenhoff, E. H. M. L. Heuberger, and B. Poolman, "Quaternary structure and function of transport proteins," *Trends in Biochemical Sciences*, vol. 27, no. 5, pp. 242–249, 2002. DOI: 10.1016/S0968-0004(02)02077-7.

[6] N. Okashah, Q. Wan, S. Ghosh, M. Sandhu, A. Inoue, N. Vaidehi, and N. A. Lambert, "Variable G protein determinants of GPCR coupling selectivity," *Proceedings of the National Academy of Sciences*, p. 201 905 993, 2019. DOI: 10.1073/pnas.1905993116.

[7] M. P. Mummadisetti, J. L. Drake, and P. G. Falkowski, "The spatial network of skeletal proteins in a stony coral," *Journal of The Royal Society Interface*, vol. 18, no. 175, p. 20 200 859, 2021.

[8] T. Will and V. Helms, "Identifying transcription factor complexes and their roles," *Bioinformatics*, vol. 30, no. 17, pp. i415–i421, 2014. DOI: 10.1093/bioinformatics/btu448.

[9] D. S. Wishart, Y. D. Feunang, A. C. Guo, E. J. Lo, A. Marcu, J. R. Grant, T. Sajed, D. Johnson, C. Li, Z. Sayeeda, *et al.*, "DrugBank 5.0: a major update to the DrugBank database for 2018," *Nucleic acids research*, vol. 46, no. D1, pp. D1074–D1082, 2018.

[10] J. J. Irwin and B. K. Shoichet, "ZINC- a free database of commercially available compounds for virtual screening," *Journal of chemical information and modeling*, vol. 45, no. 1, pp. 177–182, 2005.

[11] S. Sledzieski, R. Singh, L. Cowen, and B. Berger, "D-SCRIPT translates genome to phenome with sequence-based, structure-aware, genome-scale predictions of protein-protein interactions," *Cell Systems*, vol. 12, no. 10, pp. 969–982, 2021.

[12] R. Singh, K. Devkota, S. Sledzieski, B. Berger, and L. Cowen, "Topsy-Turvy: integrating a global view into sequence-based PPI prediction," *Bioinformatics*, vol. 38, no. Supplement_1, pp. i264–i272, 2022.

[13] S. Sledzieski, K. Devkota, R. Singh, L. Cowen, and B. Berger, "TT3D: Leveraging Pre-Computed Protein 3D Sequence Models to Predict Protein-Protein Interactions," *Bioinformatics*, btad663, 2023.

[14] R. Singh, S. Sledzieski, B. Bryson, L. Cowen, and B. Berger, "Contrastive learning in protein language space predicts interactions between drugs and protein targets," *Proceedings of the National Academy of Sciences*, vol. 120, no. 24, e2220778120, Jun. 2023, ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.2220778120.

[15] S. Sledzieski, M. Kshirsagar, M. Baek, B. Berger, R. Dodhia, and J. L. Ferres, "Democratizing protein language models with parameter-efficient fine-tuning," *bioRxiv*, 2023.

[16] J. Watson and F. Crick, "On protein synthesis," in *The Symposia of the Society for Experimental Biology*, vol. 12, 1958, pp. 138–163.

[17] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology*, vol. 48, no. 3, pp. 443–453, 1970.

[18] T. F. Smith, M. S. Waterman, *et al.*, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, 1981.

[19] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of molecular biology*, vol. 215, no. 3, pp. 403–410, 1990.

[20] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[21] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press, 1998.

[22] J. Söding, "Protein homology detection by hmm–hmm comparison," *Bioinformatics*, vol. 21, no. 7, pp. 951–960, 2005.

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

[24] P. J. Werbos, "Applications of advances in nonlinear sensitivity analysis," in *System Modeling and Optimization: Proceedings of the 10th IFIP Conference New York City, USA, August 31–September 4, 1981*, Springer, 2005, pp. 762–770.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[27] 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*, vol. 30, 2017.

[28] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[29] S. Fields and O.-k. Song, "A novel genetic system to detect protein–protein interactions," *Nature*, vol. 340, no. 6230, pp. 245–246, 1989.

[30] N. J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis, *et al.*, "Global landscape of protein complexes in the yeast Saccharomyces cerevisiae," *Nature*, vol. 440, no. 7084, pp. 637–643, 2006.

[31] A. Kumar and M. Snyder, "Protein complexes take the bait," *Nature*, vol. 415, no. 6868, pp. 123–124, 2002.

[32] N. Sahni, S. Yi, M. Taipale, J. I. F. Bass, J. Coulombe-Huntington, F. Yang, J. Peng, J. Weile, G. I. Karras, Y. Wang, *et al.*, "Widespread macromolecular interaction perturbations in human genetic disorders," *Cell*, vol. 161, no. 3, pp. 647–660, 2015.

[33] M. Taipale, G. Tucker, J. Peng, I. Krykbaeva, Z.-Y. Lin, B. Larsen, H. Choi, B. Berger, A.-C. Gingras, and S. Lindquist, "A quantitative chaperone interaction network reveals the architecture of cellular protein homeostasis pathways," *Cell*, vol. 158, no. 2, pp. 434–448, 2014.

[34] R. Sharan, I. Ulitsky, and R. Shamir, "Network-based prediction of protein function," *Molecular Systems biology*, vol. 3, no. 1, p. 88, 2007.

[35] S. Navlakha and C. Kingsford, "The power of protein interaction networks for associating genes with diseases," *Bioinformatics*, vol. 26, no. 8, pp. 1057–1063, 2010.

[36] H. Cho, B. Berger, and J. Peng, "Compact Integration of Multi-Network Topology for Functional Analysis of Genes," *Cell Systems*, vol. 3, no. 6, 540–548.e5, 2016. DOI: https://doi.org/10.1016/j.cels.2016.10.017.

[37] L. Cowen, T. Ideker, B. J. Raphael, and R. Sharan, "Network propagation: a universal amplifier of genetic associations," *Nature Reviews Genetics*, vol. 18, no. 9, p. 551, 2017.

[38] S. Choobdar, M. E. Ahsen, J. Crawford, M. Tomasoni, T. Fang, D. Lamparter, J. Lin, B. Hescott, X. Hu, J. Mercer, *et al.*, "Assessment of network module identification across complex diseases," *Nature Methods*, vol. 16, no. 9, pp. 843–852, 2019.

[39] C. Lei and J. Ruan, "A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity," *Bioinformatics*, vol. 29, no. 3, pp. 355–364, 2013.

[40] Y. Hulovatyy, R. W. Solava, and T. Milenković, "Revealing missing parts of the interactome via link prediction," *PloS one*, vol. 9, no. 3, e90073, 2014.

[41] I. A. Kovács, K. Luck, K. Spirohn, Y. Wang, C. Pollis, S. Schlabach, W. Bian, D.-K. Kim, N. Kishore, T. Hao, *et al.*, "Network-based prediction of protein interactions," *Nature communications*, vol. 10, no. 1, pp. 1–8, 2019.

[42] K. Devkota, J. M. Murphy, and L. J. Cowen, "GLIDE: combining local methods and diffusion state embeddings to predict missing interactions in biological networks," *Bioinformatics*, vol. 36, no. Supplement_1, pp. i464–i473, 2020.

[43] M. Franz, H. Rodriguez, C. Lopes, K. Zuberi, J. Montojo, G. D. Bader, and Q. Morris, "GeneMANIA update 2018," *Nucleic acids research*, vol. 46, no. W1, W60–W64, 2018.

[44] D. Szklarczyk, A. L. Gable, D. Lyon, A. Junge, S. Wyder, J. Huerta-Cepas, M. Simonovic, N. T. Doncheva, J. H. Morris, P. Bork, *et al.*, "STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets," *Nucleic acids research*, vol. 47, no. D1, pp. D607–D613, 2019.

[45] B. Wang, A. Pourshafeie, M. Zitnik, J. Zhu, C. D. Bustamante, S. Batzoglou, and J. Leskovec, "Network enhancement as a general method to denoise weighted biological networks," *Nature communications*, vol. 9, no. 1, pp. 1–8, 2018.

[46] J. Xu, "Distance-based protein folding powered by deep learning," *Proceedings of the National Academy of Sciences*, vol. 116, no. 34, pp. 16 856–16 865, 2019.

[47] M. Chen, C. J.-T. Ju, G. Zhou, X. Chen, T. Zhang, K.-W. Chang, C. Zaniolo, and W. Wang, "Multifaceted protein–protein interaction prediction based on Siamese residual RCNN," *Bioinformatics*, vol. 35, no. 14, pp. i305–i314, 2019.

[48] T. Bepler and B. Berger, "Learning the protein language: Evolution, structure, and function," *Cell Systems*, vol. 12, no. 6, pp. 654–669, 2021.

[49] H. Hwang, T. Vreven, J. Janin, and Z. Weng, "Protein–protein docking benchmark version 4.0," *Proteins: Structure, Function, and Bioinformatics*, vol. 78, no. 15, pp. 3111–3114, 2010.

[50] S. Hashemifar, B. Neyshabur, A. A. Khan, and J. Xu, "Predicting protein-protein interactions through sequence-based deep learning," in *Bioinformatics*, vol. 34, Oxford University Press, Sep. 2018, pp. i802–i810. DOI: 10.1093/bioinformatics/bty573.

[51] W. Li and A. Godzik, "CD-HIT: a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, 2006.

[52] L. Fu, B. Niu, Z. Zhu, S. Wu, and W. Li, "CD-HIT: accelerated for clustering the next-generation sequencing data," *Bioinformatics*, vol. 28, no. 23, pp. 3150–3152, 2012.

[53] T. Bepler and B. Berger, "Learning protein sequence embeddings using information from structure," *International Conference on Learning Representations (ICLR)*, 2019. arXiv: 1902.08661 [cs.LG].

[54] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, D. Guo, M. Ott, C. L. Zitnick, J. Ma, and R. Fergus, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *bioRxiv*, 2020. DOI: 10.1101/622803.

[55] Y. Luo, L. Vo, H. Ding, Y. Su, Y. Liu, W. W. Qian, H. Zhao, and J. Peng, "Evolutionary context-integrated deep sequence modeling for protein engineering," in *International Conference on Research in Computational Molecular Biology*, Springer, 2020, pp. 261–263.

[56] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," *arXiv preprint arXiv:1503.00075*, 2015.

[57] M. Cao, H. Zhang, J. Park, N. M. Daniels, M. E. Crovella, L. J. Cowen, and B. Hescott, "Going the distance for protein function prediction: a new distance metric for protein interaction networks," *PloS one*, vol. 8, no. 10, e76339, 2013.

[58] M. Cao, C. M. Pietras, X. Feng, K. J. Doroschak, T. Schaffner, J. Park, H. Zhang, L. J. Cowen, and B. J. Hescott, "New directions for diffusion-based network prediction of protein function: incorporating pathways with confidence," *Bioinformatics*, vol. 30, no. 12, pp. i219–i227, 2014.

[59] J. Thurmond, J. L. Goodman, V. B. Strelets, H. Attrill, L. S. Gramates, S. J. Marygold, B. B. Matthews, G. Millburn, G. Antonazzo, V. Trovisco, *et al.*,

"FlyBase 2.0: the next generation," *Nucleic acids research*, vol. 47, no. D1, pp. D759–D765, 2019.

[60] M. D. Adams, S. E. Celniker, R. A. Holt, C. A. Evans, J. D. Gocayne, P. G. Amanatides, S. E. Scherer, P. W. Li, R. A. Hoskins, R. F. Galle, *et al.*, "The genome sequence of Drosophila melanogaster," *Science*, vol. 287, no. 5461, pp. 2185–2195, 2000.

[61] C. Zhao and Z. Wang, "GOGO: An improved algorithm to measure the semantic similarity between gene ontology terms," *Scientific reports*, vol. 8, no. 1, pp. 1–10, 2018.

[62] S. R. Eddy, "Accelerated profile HMM searches," *PLoS computational biology*, vol. 7, no. 10, e1002195, 2011.

[63] S. Z. Alborzi, D. W. Ritchie, and M.-D. Devignes, "Computational discovery of direct associations between GO terms and protein domains," *BMC Bioinformatics*, vol. 19, no. 14, pp. 53–66, 2018.

[64] E. L. Sonnhammer, S. R. Eddy, and R. Durbin, "Pfam: a comprehensive database of protein domain families based on seed alignments," *Proteins: Structure, Function, and Bioinformatics*, vol. 28, no. 3, pp. 405–420, 1997.

[65] R. D. Finn, A. Bateman, J. Clements, P. Coggill, R. Y. Eberhardt, S. R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry, *et al.*, "Pfam: the protein families database," *Nucleic Acids Research*, vol. 42, no. D1, pp. D222–D230, 2014.

[66] U. Raudvere, L. Kolberg, I. Kuzmin, T. Arak, P. Adler, H. Peterson, and J. Vilo, "g: Profiler: a web server for functional enrichment analysis and conversions of gene lists (2019 update)," *Nucleic acids research*, vol. 47, no. W1, W191–W198, 2019.

[67] M. M. Halstead, A. Islas-Trejo, D. E. Goszczynski, J. F. Medrano, H. Zhou, and P. J. Ross, "Large-scale multiplexing permits full-length transcriptome annotation of

32 bovine tissues from a single nanopore flow cell," *Frontiers in Genetics*, vol. 12, p. 664 260, 2021.

[68] A. Alonso, J. Sasin, N. Bottini, I. Friedberg, I. Friedberg, A. Osterman, A. Godzik, T. Hunter, J. Dixon, and T. Mustelin, "Protein tyrosine phosphatases in the human genome," *Cell*, vol. 117, no. 6, pp. 699–711, 2004.

[69] A. Alonso and R. Pulido, "The extended human PTP ome: A growing tyrosine phosphatase family," *The FEBS journal*, vol. 283, no. 8, pp. 1404–1429, 2016.

[70] J. Kim, I. Kim, S. K. Han, J. U. Bowie, and S. Kim, "Network rewiring is an important mechanism of gene essentiality change," *Scientific reports*, vol. 2, p. 900, 2012.

[71] R. C. Edgar, "MUSCLE: a multiple sequence alignment method with reduced time and space complexity," *BMC bioinformatics*, vol. 5, no. 1, pp. 1–19, 2004.

[72] T. D. Schneider and R. M. Stephens, "Sequence logos: A new way to display consensus sequences," *Nucleic acids research*, vol. 18, no. 20, pp. 6097–6100, 1990.

[73] G. E. Crooks, G. Hon, J.-M. Chandonia, and S. E. Brenner, "WebLogo: A Sequence Logo Generator," *Genome Research*, vol. 14, no. 6, pp. 1188–1190, 2004. DOI: 10.1101/gr.849004.

[74] D. E. Gordon, G. M. Jang, M. Bouhaddou, J. Xu, K. Obernier, K. M. White, M. J. O'Meara, V. V. Rezelj, J. Z. Guo, D. L. Swaney, *et al.*, "A SARS-CoV-2 protein interaction map reveals targets for drug repurposing," *Nature*, pp. 1–13, 2020.

[75] G. Yu, L.-G. Wang, Y. Han, and Q.-Y. He, "clusterProfiler: an R package for comparing biological themes among gene clusters," *Omics: a journal of integrative biology*, vol. 16, no. 5, pp. 284–287, 2012.

[76] P. Mutowo, A. P. Bento, N. Dedman, A. Gaulton, A. Hersey, J. Lomax, and J. P. Overington, "A drug target slim: using gene ontology and gene ontology annotations to navigate protein-ligand target space in ChEMBL," *Journal of biomedical semantics*, vol. 7, no. 1, p. 59, 2016.

[77] R. Flamary and N. Courty, *POT Python Optimal Transport library*, 2017. [Online]. Available: https://pythonot.github.io/.

[78] F. Richoux, C. Servantie, C. Borès, and S. Téletchéa, "Comparing two deep learning sequence-based models for protein-protein interaction prediction," *bioRxiv*, 2019. arXiv: 1901.06268.

[79] J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, and H. Jiang, "Predicting protein–protein interactions based only on sequences information," *Proceedings of the National Academy of Sciences*, vol. 104, no. 11, pp. 4337–4341, 2007.

[80] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[81] Y. Jiang, M. Xie, W. Chen, R. Talbot, J. F. Maddox, T. Faraut, C. Wu, D. M. Muzny, Y. Li, W. Zhang, *et al.*, "The sheep genome illuminates biology of the rumen and lipid metabolism," *Science*, vol. 344, no. 6188, pp. 1168–1173, 2014.

[82] J.-T. Yu, Y. Liu, P. Dong, R.-E. Cheng, S.-X. Ke, K.-Q. Chen, J.-J. Wang, Z.-S. Shen, Q.-Y. Tang, and Z. Zhang, "Up-regulation of antioxidative proteins TRX1, TXNL1 and TXNRD1 in the cortex of PTZ kindling seizure model mice," *PloS one*, vol. 14, no. 1, e0210670, 2019.

[83] J.-M. Zhao and T.-G. Qi, "The role of TXNL1 in disease: treatment strategies for cancer and diseases with oxidative stress," *Molecular Biology Reports*, pp. 1–6, 2021.

[84] N. Lee, D.-K. Kim, S. H. Han, H. G. Ryu, S. J. Park, K.-T. Kim, and K. Y. Choi, "Comparative interactomes of VRK1 and VRK3 with their distinct roles in the cell cycle of liver cancer," *Molecules and cells*, vol. 40, no. 9, p. 621, 2017.

[85] B. Brauksiepe, A. O. Mujica, H. Herrmann, and E. R. Schmidt, "The Serine/threonine kinase Stk33 exhibits autophosphorylation and phosphorylates the intermediate filament protein Vimentin," *BMC biochemistry*, vol. 9, no. 1, pp. 1–12, 2008.

[86] H. Schwarzenbacher, J. Burgstaller, F. R. Seefried, C. Wurmser, M. Hilbe, S. Jung, C. Fuerst, N. Dinhopl, H. Weissenböck, B. Fuerst-Waltl, *et al.*, "A missense mutation in TUBD1 is associated with high juvenile mortality in Braunvieh and Fleckvieh cattle," *BMC genomics*, vol. 17, no. 1, pp. 1–13, 2016.

[87] K. Sasaki, N. Yamagishi, K. Kizaki, B. Devkota, and K. Hashizume, "Microarray-based gene expression profiling of peripheral blood mononuclear cells in dairy cows with experimental hypocalcemia and milk fever," *Journal of dairy science*, vol. 97, no. 1, pp. 247–258, 2014.

[88] M. V. Dodson, G. J. Hausman, L. Guan, M. Du, T. P. Rasmussen, S. P. Poulos, P. Mir, W. G. Bergen, M. E. Fernyhough, D. C. McFarland, *et al.*, "Lipid metabolism, adipocyte depot physiology and utilization of meat animals as experimental models for metabolic research," *International Journal of Biological Sciences*, vol. 6, no. 7, p. 691, 2010.

[89] J. R. Aschenbach, N. B. Kristensen, S. S. Donkin, H. M. Hammon, and G. B. Penner, "Gluconeogenesis in dairy cows: the secret of making sweet milk from sour dough," *IUBMB life*, vol. 62, no. 12, pp. 869–877, 2010.

[90] L. F. C. Castro, M. Lopes-Marques, J. M. Wilson, E. Rocha, M. A. Reis-Henriques, M. M. Santos, and I. Cunha, "A novel Acetyl-CoA synthetase short-chain subfamily member 1 (Acss1) gene indicates a dynamic history of paralogue retention and loss in vertebrates," *Gene*, vol. 497, no. 2, pp. 249–255, 2012.

[91] R. N. Rodionov, N. Jarzebska, N. Weiss, and S. R. Lentz, "AGXT2: a promiscuous aminotransferase," *Trends in pharmacological sciences*, vol. 35, no. 11, pp. 575–582, 2014.

[92] S. Soma, M. N. Morgada, M. T. Naik, A. Boulet, A. A. Roesler, N. Dziuba, A. Ghosh, Q. Yu, P. A. Lindahl, J. B. Ames, *et al.*, "COA6 is structurally tuned to function as a thiol-disulfide oxidoreductase in copper delivery to mitochondrial cytochrome c oxidase," *Cell reports*, vol. 29, no. 12, pp. 4114–4126, 2019.

[93] S. Kamiński, P. Brym, and E. Wójcik, "A note on associations between polymorphism with-in the 2, 4-dienoyl-CoA reductase gene (DECR1) and growth rate of Polish Landrace boars," *Journal of Animal and Feed Sciences*, vol. 18, no. 1, pp. 71–77, 2009.

[94] Z. Zhu and G. K. K. Leung, "More Than a Metabolic Enzyme: MTHFD2 as a Novel Target for Anticancer Therapy?" *Frontiers in oncology*, vol. 10, p. 658, 2020.

[95] A. Ginguay, L. Cynober, E. Curis, and I. Nicolis, "Ornithine aminotransferase, an important glutamate-metabolizing enzyme at the crossroads of multiple metabolic pathways," *Biology*, vol. 6, no. 1, p. 18, 2017.

[96] A. Gohla, "Do metabolic HAD phosphatases moonlight as protein phosphatases?" *Biochimica et Biophysica Acta (BBA)-Molecular Cell Research*, vol. 1866, no. 1, pp. 153–166, 2019.

[97] V. Reiter, D. M. Matschkal, M. Wagner, D. Globisch, A. C. Kneuttinger, M. Müller, and T. Carell, "The CDK5 repressor CDK5RAP1 is a methylthiotransferase acting on nuclear and mitochondrial RNA," *Nucleic acids research*, vol. 40, no. 13, pp. 6235–6240, 2012.

[98] D. Ghezzi, E. Baruffini, T. B. Haack, F. Invernizzi, L. Melchionda, C. Dallabona, T. M. Strom, R. Parini, A. B. Burlina, T. Meitinger, *et al.*, "Mutations of the mitochondrial-tRNA modifier MTO1 cause hypertrophic cardiomyopathy and lactic

acidosis," *The American Journal of Human Genetics*, vol. 90, no. 6, pp. 1079–1087, 2012.

[99] Y. Chen, Z.-R. Ruan, Y. Wang, Q. Huang, M.-Q. Xue, X.-L. Zhou, and E.-D. Wang, "A threonyl-tRNA synthetase-like protein has tRNA aminoacylation and editing activities," *Nucleic acids research*, vol. 46, no. 7, pp. 3643–3656, 2018.

[100] J. L. Pohjoismäki, S. Wanrooij, A. K. Hyvärinen, S. Goffart, I. J. Holt, J. N. Spelbrink, and H. T. Jacobs, "Alterations to the expression level of mitochondrial transcription factor A, TFAM, modify the mode of mitochondrial DNA replication in cultured human cells," *Nucleic acids research*, vol. 34, no. 20, pp. 5815–5828, 2006.

[101] A. Besse, P. Wu, F. Bruni, T. Donti, B. H. Graham, W. J. Craigen, R. McFarland, P. Moretti, S. Lalani, K. L. Scott, *et al.*, "The GABA transaminase, ABAT, is essential for mitochondrial nucleoside metabolism," *Cell metabolism*, vol. 21, no. 3, pp. 417–427, 2015.

[102] E. Fassone, A. J. Duncan, J.-W. Taanman, A. T. Pagnamenta, M. I. Sadowski, T. Holand, W. Qasim, P. Rutland, S. E. Calvo, V. K. Mootha, *et al.*, "FOXRED1, encoding an FAD-dependent oxidoreductase complex-I-specific molecular chaperone, is mutated in infantile-onset mitochondrial encephalopathy," *Human molecular genetics*, vol. 19, no. 24, pp. 4837–4847, 2010.

[103] A. Srivastava, K. R. Srivastava, M. Hebbar, C. Galada, R. Kadavigrere, F. Su, X. Cao, A. M. Chinnaiyan, K. M. Girisha, A. Shukla, *et al.*, "Genetic diversity of NDUFV1-dependent mitochondrial complex I deficiency," *European Journal of Human Genetics*, vol. 26, no. 11, pp. 1582–1587, 2018.

[104] X. Gao and M. Oba, "Characteristics of dairy cows with a greater or lower risk of subacute ruminal acidosis: volatile fatty acid absorption, rumen digestion, and

expression of genes in rumen epithelial cells," *Journal of dairy science*, vol. 99, no. 11, pp. 8733–8745, 2016.

[105] M. Zarrin, O. Wellnitz, H. A. van Dorland, J. J. Gross, and R. Bruckmaier, "Hyperketonemia during lipopolysaccharide-induced mastitis affects systemic and local intramammary metabolism in dairy cows," *Journal of Dairy Science*, vol. 97, no. 6, pp. 3531–3541, 2014.

[106] P. Yadav, P. Kumar, M. Mukesh, R. Kataria, A. Yadav, A. Mohanty, and B. Mishra, "Kinetics of lipogenic genes expression in milk purified mammary epithelial cells (MEC) across lactation and their correlation with milk and fat yield in buffalo," *Research in veterinary science*, vol. 99, pp. 129–136, 2015.

[107] L. Jing, C. Yang, L. Guiying, *et al.*, "Relationship between the Polymorphisms of DECR1 Gene and Meat Quality Traits in Yanbian Yellow Cattle [J]," *Journal of Anhui Agricultural Sciences*, vol. 34, p. 20, 2009.

[108] H. Huang, J. Cao, G. Guo, X. Li, Y. Wang, Y. Yu, S. Zhang, Q. Zhang, and Y. Zhang, "Genome-wide association study identifies QTLs for displacement of abomasum in Chinese Holstein cattle," *Journal of animal science*, vol. 97, no. 3, pp. 1133–1142, 2019.

[109] X. Wei, Y. Zhang, Z. Fu, and L. Zhang, "The association between polymorphisms in the MRPL4 and TNF-α genes and susceptibility to allergic rhinitis," *PLoS One*, vol. 8, no. 3, e57981, 2013.

[110] C. W. Heizmann, "S100 proteins: Diagnostic and prognostic biomarkers in laboratory medicine," *Biochimica et Biophysica Acta (BBA)-Molecular Cell Research*, vol. 1866, no. 7, pp. 1197–1206, 2019.

[111] K. Hsu, C. Champaiboon, B. D. Guenther, B. S. Sorenson, A. Khammanivong, K. F. Ross, C. L. Geczy, and M. C. Herzberg, "Anti-infective protective properties of S100 calgranulins," *Anti-Inflammatory & Anti-Allergy Agents in Medicinal Chemistry*

(Formerly Current Medicinal Chemistry-Anti-Inflammatory and Anti-Allergy Agents), vol. 8, no. 4, pp. 290–305, 2009.

[112] S. E. Permyakov, E. N. Yundina, A. S. Kazakov, M. E. Permyakova, V. N. Uversky, and E. A. Permyakov, "Mouse S100G protein exhibits properties characteristic of a calcium sensor," *Cell calcium*, vol. 87, p. 102 185, 2020.

[113] Y. Shi, C. He, C. Ma, T. Yu, Y. Cong, W. Cai, and Z. Liu, "Smad nuclear interacting protein 1 (SNIP1) inhibits intestinal inflammation through regulation of epithelial barrier function," *Mucosal immunology*, vol. 11, no. 3, pp. 835–845, 2018.

[114] N. Rodríguez, A. Peláez, R. Barderas, and G. Domínguez, "Clinical implications of the deregulated TP73 isoforms expression in cancer," *Clinical and Translational Oncology*, vol. 20, no. 7, pp. 827–836, 2018.

[115] J. Guo, Z. Fu, J. Wei, W. Lu, J. Feng, and S. Zhang, "PRRX1 promotes epithelial–mesenchymal transition through the Wnt/β-catenin pathway in gastric cancer," *Medical oncology*, vol. 32, no. 1, p. 393, 2015.

[116] X. Yang, Y. Lin, Y. Shi, B. Li, W. Liu, W. Yin, Y. Dang, Y. Chu, J. Fan, and R. He, "FAP promotes immunosuppression by cancer-associated fibroblasts in the tumor microenvironment via STAT3–CCL2 signaling," *Cancer research*, vol. 76, no. 14, pp. 4124–4135, 2016.

[117] M. Garcia, B. Bradford, and T. Nagaraja, "Invited review: ruminal microbes, microbial products, and systemic inflammation," *The Professional Animal Scientist*, vol. 33, no. 6, pp. 635–650, 2017.

[118] M. L. Turner, J. G. Cronin, P. G. Noleto, and I. M. Sheldon, "Glucose availability and AMP-activated protein kinase link energy metabolism and innate immunity in the bovine endometrium," *PloS one*, vol. 11, no. 3, e0151416, 2016.

[119] S. J. Kim and S. E. Ryu, "Structure and catalytic mechanism of human protein tyrosine phosphatome," *BMB reports*, vol. 45, no. 12, p. 693, 2012.

[120] A. Ingles-Prieto, B. Ibarra-Molero, A. Delgado-Delgado, R. Perez-Jimenez, J. M. Fernandez, E. A. Gaucher, J. M. Sanchez-Ruiz, and J. A. Gavira, "Conservation of protein structure over four billion years," *Structure*, vol. 21, no. 9, pp. 1690–1697, 2013.

[121] Y. Luo, J. Peng, and J. Ma, "When causal inference meets deep learning," *Nature Machine Intelligence*, vol. 2, no. 8, pp. 426–427, 2020.

[122] B. Hie, B. D. Bryson, and B. A. Berger, "Leveraging Uncertainty in Machine Learning Accelerates Biological Discovery and Design," *Cell Systems*, 2020.

[123] D. S. Marks, L. J. Colwell, R. Sheridan, T. A. Hopf, A. Pagnani, R. Zecchina, and C. Sander, "Protein 3D structure computed from evolutionary sequence variation," *PloS one*, vol. 6, no. 12, e28766, 2011.

[124] Y. Liu, P. Palmedo, Q. Ye, B. Berger, and J. Peng, "Enhancing evolutionary couplings with deep convolutional neural networks," *Cell systems*, vol. 6, no. 1, pp. 65–74, 2018.

[125] H. Kamisetty, S. Ovchinnikov, and D. Baker, "Assessing the utility of coevolution-based residue–residue contact predictions in a sequence-and structure-rich era," *Proceedings of the National Academy of Sciences*, vol. 110, no. 39, pp. 15 674–15 679, 2013.

[126] H. Zeng, S. Wang, T. Zhou, F. Zhao, X. Li, Q. Wu, and J. Xu, "ComplexContact: a web server for inter-protein contact prediction using deep learning," *Nucleic acids research*, vol. 46, no. W1, W432–W437, 2018.

[127] T. A. Hopf, C. P. Schärfe, J. P. Rodrigues, A. G. Green, O. Kohlbacher, C. Sander, A. M. Bonvin, and D. S. Marks, "Sequence co-evolution gives 3D contacts and structures of protein complexes," *Elife*, vol. 3, e03430, 2014.

[128] A. G. Green, H. Elhabashy, K. P. Brock, R. Maddamsetti, O. Kohlbacher, and D. S. Marks, "Large-scale discovery of protein interactions at residue resolution using

co-evolution calculated from genomic sequences," *Nature communications*, vol. 12, no. 1, pp. 1–12, 2021.

[129] Q. Cong, I. Anishchenko, S. Ovchinnikov, and D. Baker, "Protein interaction networks revealed by proteome coevolution," *Science*, vol. 365, no. 6449, pp. 185–189, 2019.

[130] R. Singh, J. Xu, and B. Berger, "Struct2net: integrating structure into protein-protein interaction prediction," in *Biocomputing 2006*, World Scientific, 2006, pp. 403–414.

[131] R. Singh, D. Park, J. Xu, R. Hosur, and B. Berger, "Struct2Net: a web service to predict protein–protein interactions using a structure-based approach," *Nucleic acids research*, vol. 38, no. suppl_2, W508–W515, 2010.

[132] R. Hosur, J. Xu, J. Bienkowska, and B. Berger, "iWRAP: an interface threading approach with application to prediction of cancer-related protein–protein interactions," *Journal of molecular biology*, vol. 405, no. 5, pp. 1295–1310, 2011.

[133] 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*, vol. 596, no. 7873, pp. 583–589, 2021.

[134] M. van Kempen, S. Kim, C. Tumescheit, M. Mirdita, J. Söding, and M. Steinegger, "Foldseek: fast and accurate protein structure search," *bioRxiv*, 2022. DOI: 10.1101/2022.02.07.479398.

[135] Q. C. Zhang, D. Petrey, L. Deng, L. Qiang, Y. Shi, C. A. Thu, B. Bisikirska, C. Lefebvre, D. Accili, T. Hunter, *et al.*, "Structure-based prediction of protein–protein interactions on a genome-wide scale," *Nature*, vol. 490, no. 7421, pp. 556–560, 2012.

[136] W. Hamilton, P. Bajaj, M. Zitnik, D. Jurafsky, and J. Leskovec, "Embedding Logical Queries on Knowledge Graphs," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[137] M. Coskun and M. Koyutürk, "Node Similarity Based Graph Convolution for Link Prediction in Biological Networks," *Bioinformatics*, vol. 37, no. 23, pp. 4501–4508, 2021.

[138] K. Huang, C. Xiao, L. M. Glass, M. Zitnik, and J. Sun, "SkipGNN: predicting molecular interactions with skip-graph networks," *Scientific Reports*, vol. 10, no. 1, pp. 1–16, 2020.

[139] H. Y. Yuen and J. Jansson, "Better Link Prediction for Protein-Protein Interaction Networks," in *2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE)*, IEEE, 2020, pp. 53–60.

[140] L. Kumar, N. Brenner, S. Sledzieski, *et al.*, "Transfer of Knowledge from Model Organisms to Evolutionarily Distant Non-Model Organisms: The Coral Pocillopora damicornis Membrane Signaling Receptome," *bioRxiv*, 2021.

[141] 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*, pp. 2021–10, 2021.

[142] A. Grover and J. Leskovec, "Node2vec: Scalable Feature Learning for Networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 855–864, ISBN: 9781450342322. DOI: 10.1145/2939672.2939754.

[143] F. Yang, K. Fan, D. Song, and H. Lin, "Graph-based prediction of Protein-protein interactions with attributed signed graph embedding," *BMC Bioinformatics*, vol. 21, no. 1, p. 323, Jul. 2020.

[144] H. Carter, M. Hofree, and T. Ideker, "Genotype to phenotype via network analysis," *Current opinion in genetics & development*, vol. 23, no. 6, pp. 611–621, 2013.

[145] K. A. Porter, I. Desta, D. Kozakov, and S. Vajda, "What method to use for protein–protein docking?" *Current Opinion in Structural Biology*, vol. 55, pp. 1–7, 2019.

[146] 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*, 2022.

[147] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, *et al.*, "Evolutionary-scale prediction of atomic-level protein structure with a language model," *Science*, vol. 379, no. 6637, pp. 1123–1130, 2023.

[148] M. Baek, F. DiMaio, I. Anishchenko, J. Dauparas, S. Ovchinnikov, G. R. Lee, J. Wang, Q. Cong, L. N. Kinch, R. D. Schaeffer, *et al.*, "Accurate prediction of protein structures and interactions using a three-track neural network," *Science*, vol. 373, no. 6557, pp. 871–876, 2021.

[149] M. Varadi, S. Anyango, M. Deshpande, S. Nair, C. Natassia, *et al.*, "AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models," *Nucleic acids research*, vol. 50, no. D1, pp. D439–D444, 2022.

[150] S. K. Burley, C. Bhikadiya, C. Bi, S. Bittrich, H. Chao, *et al.*, "RCSB Protein Data Bank (RCSB.org): delivery of experimentally-determined PDB structures alongside one million computed structure models of proteins from artificial intelligence/machine learning," *Nucleic Acids Research*, vol. 51, no. D1, pp. D488–D508, 2023.

[151] W. Zhu, A. Shenoy, P. Kundrotas, and A. Elofsson, "Evaluation of AlphaFold-Multimer prediction on multi-chain protein complexes," *Bioinformatics*, vol. 39, no. 7, btad424, 2023.

[152] K. Weißenow, M. Heinzinger, and B. Rost, "Protein language-model embeddings for fast, accurate, and alignment-free protein structure prediction," *Structure*, vol. 30, no. 8, pp. 1169–1177, 2022.

[153] D. F. Burke, P. Bryant, I. Barrio-Hernandez, D. Memon, G. Pozzati, A. Shenoy, W. Zhu, A. S. Dunham, P. Albanese, A. Keller, *et al.*, "Towards a structurally resolved human protein interaction network," *Nature Structural & Molecular Biology*, vol. 30, no. 2, pp. 216–225, 2023.

[154] M. Lensink, G. Brysbaert, N. Raouraoua, P. Bates, M. Giulini, *et al.*, "Impact of AlphaFold on Structure Prediction of Protein Complexes: The CASP15-CAPRI Experiment," *Authorea Preprints*, 2023.

[155] K. Harini, C. Christoffer, M. M. Gromiha, and D. Kihara, "Pairwise and Multi-chain Protein Docking Enhanced Using LZerD Web Server," in *Protein-Protein Interactions: Methods and Protocols*, Springer, 2023, pp. 355–373.

[156] M. van Kempen, S. S. Kim, C. Tumescheit, M. Mirdita, J. Lee, C. L. Gilchrist, J. Söding, and M. Steinegger, "Fast and accurate protein structure search with Foldseek," *Nature Biotechnology*, pp. 1–4, 2023.

[157] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[158] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic acids research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[159] M. Steinegger and J. Söding, "MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets," *Nature Biotechnology*, vol. 35, no. 11, pp. 1026–1028, 2017.

[160] I. Barrio-Hernandez, J. Yeo, J. Jänes, T. Wein, M. Varadi, S. Velankar, P. Beltrao, and M. Steinegger, "Clustering predicted structures at the scale of the known protein universe," *bioRxiv*, pp. 2023–03, 2023.

[161] A. Abid, A. Abdalla, A. Abid, D. Khan, A. Alfozan, and J. Zou, *Gradio: Hassle-free sharing and testing of ML models in the wild*, Jun. 2019. DOI: 10.48550/arXiv.1906.02569.

[162] M. Heinzinger, K. Weissenow, J. G. Sanchez, A. Henkel, M. Steinegger, and B. Rost, "ProstT5: Bilingual Language Model for Protein Sequence and Structure," *bioRxiv*, pp. 2023–07, 2023.

[163] D. Szklarczyk, A. L. Gable, K. C. Nastou, *et al.*, "The STRING database in 2021: customizable protein-protein networks, and functional characterization of user-uploaded gene/measurement sets," *Nucleic Acids Res*, vol. 49, no. D1, pp. D605–D612, Jan. 2021.

[164] L. Zhang, G. Yu, M. Guo, and J. Wang, "Predicting protein-protein interactions using high-quality non-interacting pairs," *BMC Bioinformatics*, vol. 19, Dec. 2018. DOI: 10.1186/s12859-018-2525-3.

[165] A. Fox, D. Taylor, and D. K. Slonim, "High throughput interaction data reveals degree conservation of hub proteins," eng, *Pacific Symposium on Biocomputing*, pp. 391–402, 2009, ISSN: 2335-6928. DOI: 10.1142/9789812836939_0037.

[166] K. R. Brown and I. Jurisica, "Unequal evolutionary conservation of human protein interactions in interologous networks," *Genome Biology*, vol. 8, no. 5, R95, May 2007, ISSN: 1474-760X. DOI: 10.1186/gb-2007-8-5-r95.

[167] K. Luck, D.-K. Kim, L. Lambourne, K. Spirohn, B. E. Begg, W. Bian, R. Brignall, T. Cafarelli, F. J. Campos-Laborie, B. Charloteaux, *et al.*, "A reference map of the human binary protein interactome," *Nature*, vol. 580, no. 7803, pp. 402–408, 2020.

[168] A. Lopes, S. Sacquin-Mora, V. Dimitrova, E. Laine, Y. Ponty, and A. Carbone, "Protein-protein interactions in a crowded environment: an analysis via cross-docking simulations and evolutionary information," *PLoS Comput Biol*, vol. 9, no. 12, e1003369, 2013.

[169] C. Dequeker, Y. Mohseni Behbahani, L. David, E. Laine, and A. Carbone, "From complete cross-docking to partners identification and binding sites predictions," *PLoS Comput Biol*, vol. 18, no. 1, e1009825, Jan. 2022.

[170] D. Smedley, S. Haider, B. Ballester, R. Holland, D. London, G. Thorisson, and A. Kasprzyk, "BioMart–biological queries made easy," *BMC Genomics*, vol. 10, no. 1, pp. 1–12, 2009.

[171] L. Pray, "Eukaryotic genome complexity," *Nature Education*, vol. 1, no. 1, p. 96, 2008.

[172] M. H. Serres, S. Gopal, L. A. Nahum, P. Liang, T. Gaasterland, and M. Riley, "A functional update of the Escherichia coli K-12 genome," *Genome Biology*, vol. 2, no. 9, pp. 1–7, 2001.

[173] R. Singh, J. Xu, and B. Berger, "Pairwise global alignment of protein interaction networks by matching neighborhood topology," in *RECOMB*, Springer, 2007, pp. 16–31.

[174] D. Ghersi and M. Singh, "Interaction-based discovery of functionally important genes in cancers," *Nucleic acids research*, vol. 42, no. 3, e18–e18, 2014.

[175] I. Budowski-Tal, R. Kolodny, and Y. Mandel-Gutfreund, "A Novel Geometry-Based Approach to Infer Protein Interface Similarity," *Scientific Reports*, vol. 8, no. 1, pp. 1–10, 2018.

[176] P. Gainza, F. Sverrisson, F. Monti, E. Rodola, D. Boscaini, M. Bronstein, and B. Correia, "Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning," *Nature Methods*, vol. 17, no. 2, pp. 184–192, 2020.

[177] L. Pinzi and G. Rastelli, "Molecular docking: shifting paradigms in drug discovery," *International journal of molecular sciences*, vol. 20, no. 18, p. 4331, 2019.

[178] B. M. Bonk, Y. Tarasova, M. A. Hicks, B. Tidor, and K. L. Prather, "Rational design of thiolase substrate specificity for metabolic engineering applications," *Biotechnology and bioengineering*, vol. 115, no. 9, pp. 2167–2182, 2018.

[179] R. C. de Melo-Minardi, K. Bastard, and F. Artiguenave, "Identification of subfamily-specific sites based on active sites modeling and clustering," *Bioinformatics*, vol. 26, no. 24, pp. 3075–3082, 2010.

[180] S. J. Trudeau, H. Hwang, D. Mathur, K. Begum, D. Petrey, D. Murray, and B. Honig, "PrePCI: A structure-and chemical similarity-informed database of predicted protein compound interactions," *bioRxiv*, 2022.

[181] E. Anderson, G. D. Veith, and D. Weininger, *SMILES, a line notation and computerized interpreter for chemical structures*. US Environmental Protection Agency, Environmental Research Laboratory, 1987.

[182] M. Bagherian, E. Sabeti, K. Wang, M. A. Sartor, Z. Nikolovska-Coleska, and K. Najarian, "Machine learning approaches and databases for prediction of drug–target interaction: a survey paper," en, *Briefings in Bioinformatics*, p. 23, 2021.

[183] B. Hie, H. Cho, and B. Berger, "Realizing private and practical pharmacological collaboration," *Science*, vol. 362, no. 6412, pp. 347–350, 2018.

[184] I. Lee, J. Keum, and H. Nam, "DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences," *PLoS computational biology*, vol. 15, no. 6, e1007129, 2019.

[185] K. Huang, C. Xiao, L. M. Glass, and J. Sun, "MolTrans: molecular interaction transformer for drug–target interaction prediction," *Bioinformatics*, vol. 37, no. 6, pp. 830–836, 2021.

[186] S. Sledzieski, R. Singh, L. Cowen, and B. Berger, "Adapting protein language models for rapid DTI prediction," *Machine Learning for Structural Biology Workshop (MLSB) at NeurIPS*, 2021.

[187] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, *et al.*, "On the Opportunities and Risks of Foundation Models," *arXiv preprint arXiv:2108.07258*, 2021.

[188] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, "Don't stop pretraining: adapt language models to domains and tasks," *arXiv preprint arXiv:2004.10964*, 2020.

[189] D. Bausch-Fluck, U. Goldmann, S. Müller, M. van Oostrum, M. Müller, O. T. Schubert, and B. Wollscheid, "The in silico human surfaceome," *Proceedings of the National Academy of Sciences*, vol. 115, no. 46, E10988–E10997, 2018.

[190] D. Mendez, A. Gaulton, A. P. Bento, J. Chambers, M. De Veij, E. Félix, M. P. Magariños, J. F. Mosquera, P. Mutowo, M. Nowotka, *et al.*, "ChEMBL: towards direct deposition of bioassay data," *Nucleic acids research*, vol. 47, no. D1, pp. D930–D940, 2019.

[191] K. Huang, T. Fu, W. Gao, Y. Zhao, Y. Roohani, J. Leskovec, C. W. Coley, C. Xiao, J. Sun, and M. Zitnik, "Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development," *arXiv preprint arXiv:2102.09548*, 2021.

[192] N. Zong, N. Li, A. Wen, V. Ngo, Y. Yu, M. Huang, S. Chowdhury, C. Jiang, S. Fu, R. Weinshilboum, *et al.*, "BETA: a comprehensive benchmark for computational drug–target prediction," *Briefings in Bioinformatics*, 2022.

[193] H. M. Dönertaş, M. Fuentealba Valenzuela, L. Partridge, and J. M. Thornton, "Gene expression-based drug repurposing to target aging," *Aging cell*, vol. 17, no. 5, e12819, 2018.

[194] D. Morselli Gysi, Í. Do Valle, M. Zitnik, A. Ameli, X. Gan, O. Varol, S. D. Ghiassian, J. Patten, R. A. Davey, J. Loscalzo, *et al.*, "Network medicine framework for identifying drug-repurposing opportunities for COVID-19," *Proceedings of the National Academy of Sciences*, vol. 118, no. 19, e2025581118, 2021.

[195] K. Huang, T. Fu, L. M. Glass, M. Zitnik, C. Xiao, and J. Sun, "DeepPurpose: a deep learning library for drug–target interaction prediction," *Bioinformatics*, vol. 36, no. 22-23, pp. 5545–5547, 2020.

[196] Y. Huang, M. Furuno, T. Arakawa, S. Takizawa, M. de Hoon, H. Suzuki, and E. Arner, "A framework for identification of on-and off-target transcriptional responses to drug treatment," *Scientific reports*, vol. 9, no. 1, pp. 1–9, 2019.

[197] S. Goldman, R. Das, K. K. Yang, and C. W. Coley, "Machine learning modeling of family wide enzyme-substrate specificity screens," *PLoS computational biology*, vol. 18, no. 2, e1009853, 2022.

[198] I. Gulrajani and D. Lopez-Paz, "In search of lost domain generalization," *arXiv preprint arXiv:2007.01434*, 2020.

[199] M. Zitnik, R. Sosič, S. Maheshwari, and J. Leskovec, *BioSNAP Datasets: Stanford Biomedical Network Dataset Collection*, http://snap.stanford.edu/biodata, Aug. 2018.

[200] T. Liu, Y. Lin, X. Wen, R. N. Jorissen, and M. K. Gilson, "BindingDB: a web-accessible database of experimentally determined protein–ligand binding affinities," *Nucleic acids research*, vol. 35, no. suppl_1, pp. D198–D201, 2007.

[201] M. I. Davis, J. P. Hunt, S. Herrgard, P. Ciceri, L. M. Wodicka, G. Pallares, M. Hocker, D. K. Treiber, and P. P. Zarrinkar, "Comprehensive analysis of kinase inhibitor selectivity," *Nature biotechnology*, vol. 29, no. 11, pp. 1046–1051, 2011.

[202] H. Huang, C. Pandya, C. Liu, N. F. Al-Obaidi, M. Wang, L. Zheng, S. Toews Keating, M. Aono, J. D. Love, B. Evans, *et al.*, "Panoramic view of a

superfamily of phosphatases through substrate profiling," *Proceedings of the National Academy of Sciences*, vol. 112, no. 16, E1974–E1983, 2015.

[203] M. Martínez-Martínez, C. Coscolín, G. Santiago, J. Chow, P. J. Stogios, R. Bargiela, C. Gertler, J. Navarro-Fernández, A. Bollinger, S. Thies, *et al.*, "Determinants and prediction of esterase substrate promiscuity patterns," *ACS chemical biology*, vol. 13, no. 1, pp. 225–234, 2017.

[204] M. Yang, C. Fehl, K. V. Lees, E.-K. Lim, W. A. Offen, G. J. Davies, D. J. Bowles, M. G. Davidson, S. J. Roberts, and B. G. Davis, "Functional and informatics analysis enables glycosyltransferase activity prediction," *Nature chemical biology*, vol. 14, no. 12, pp. 1109–1117, 2018.

[205] B. F. Fisher, H. M. Snodgrass, K. A. Jones, M. C. Andorfer, and J. C. Lewis, "Site-selective C–H halogenation using flavin-dependent halogenases identified via family-wide activity profiling," *ACS central science*, vol. 5, no. 11, pp. 1844–1856, 2019.

[206] K. Bastard, A. A. T. Smith, C. Vergne-Vaxelaire, A. Perret, A. Zaparucha, D. Melo-Minardi, A. Mariage, M. Boutard, A. Debard, C. Lechaplais, *et al.*, "Revealing the hidden functional diversity of an enzyme family," *Nature chemical biology*, vol. 10, no. 1, pp. 42–49, 2014.

[207] M. M. Mysinger, M. Carchia, J. J. Irwin, and B. K. Shoichet, "Directory of useful decoys, enhanced (DUD-E): better ligands and decoys for better benchmarking," *Journal of medicinal chemistry*, vol. 55, no. 14, pp. 6582–6594, 2012.

[208] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C. L. Zitnick, J. Ma, *et al.*, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *Proceedings of the National Academy of Sciences*, vol. 118, no. 15, 2021.

[209] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, *et al.*, "ProtTrans: Toward understanding the language of life through self-supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 10, pp. 7112–7127, 2021.

[210] H. L. Morgan, "The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service.," *Journal of Chemical Documentation*, vol. 5, no. 2, pp. 107–113, 1965.

[211] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of chemical information and modeling*, vol. 50, no. 5, pp. 742–754, 2010.

[212] S. Jaeger, S. Fulle, and S. Turk, "Mol2vec: unsupervised machine learning approach with chemical intuition," *Journal of chemical information and modeling*, vol. 58, no. 1, pp. 27–35, 2018.

[213] H. Wang, W. Li, X. Jin, K. Cho, H. Ji, J. Han, and M. D. Burke, "Chemical-reaction-aware molecule representation learning," *arXiv preprint arXiv:2109.09888*, 2021.

[214] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[215] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[216] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[217] M. Tsubaki, K. Tomii, and J. Sese, "Compound–protein interaction prediction with end-to-end learning of neural networks for graphs and sequences," *Bioinformatics*, vol. 35, no. 2, pp. 309–318, 2019.

[218] A. Scaiewicz and M. Levitt, "The language of the protein universe," *Current opinion in genetics & development*, pp. 50–56, 2015.

[219] M. Heinzinger, A. Elnaggar, Y. Wang, C. Dallago, D. Nechaev, F. Matthes, and B. Rost, "Modeling aspects of the language of life through transfer-learning protein sequences," *BMC bioinformatics*, vol. 20, no. 1, pp. 1–17, 2019.

[220] M. Heinzinger, M. Littmann, I. Sillitoe, N. Bordin, C. Orengo, and B. Rost, "Contrastive learning on protein embeddings enlightens midnight zone," *NAR genomics and bioinformatics*, vol. 4, no. 2, lqac043, 2022.

[221] F. Carles, S. Bourg, C. Meyer, and P. Bonnet, "PKIDB: A curated, annotated and updated database of protein kinase inhibitors in clinical trials," *Molecules*, vol. 23, no. 4, p. 908, 2018.

[222] W. B. Weglicki, J. H. Kramer, C. F. Spurney, J. J. Chmielinska, and I. T. Mak, "The EGFR tyrosine kinase inhibitor tyrphostin AG-1478 causes hypomagnesemia and cardiac dysfunction," *Canadian journal of physiology and pharmacology*, vol. 90, no. 8, pp. 1145–1149, 2012.

[223] R. Sordella, D. W. Bell, D. A. Haber, and J. Settleman, "Gefitinib-sensitizing EGFR mutations in lung cancer activate anti-apoptotic pathways," *Science*, vol. 305, no. 5687, pp. 1163–1167, 2004.

[224] G. J. Roth, R. Binder, F. Colbatzky, C. Dallinger, R. Schlenker-Herceg, F. Hilberg, S.-L. Wollin, and R. Kaiser, "Nintedanib: from discovery to the clinic," *Journal of medicinal chemistry*, vol. 58, no. 3, pp. 1053–1063, 2015.

[225] E. S. Wang, K. Yee, L. P. Koh, D. Hogge, S. Enschede, D. M. Carlson, M. Dudley, K. Glaser, E. McKeegan, D. H. Albert, *et al.*, "Phase 1 trial of linifanib (ABT-869) in patients with refractory or relapsed acute myeloid leukemia," *Leukemia & Lymphoma*, vol. 53, no. 8, pp. 1543–1551, 2012.

[226] N. C. Wolff, D. R. Veach, W. P. Tong, W. G. Bornmann, B. Clarkson, and R. L. Ilaria Jr, "PD166326, a novel tyrosine kinase inhibitor, has greater antileukemic activity than imatinib mesylate in a murine model of chronic myeloid leukemia," *Blood*, vol. 105, no. 10, pp. 3995–4003, 2005.

[227] V. Cibert-Goton, G. Yuan, A. Battaglia, S. Fredriksson, M. Henkemeyer, T. Sears, and I. Gavazzi, "Involvement of EphB1 receptors signalling in models of inflammatory and neuropathic pain," *PLoS One*, vol. 8, no. 1, e53673, 2013.

[228] S. Liu, W.-T. Liu, Y.-P. Liu, H.-L. Dong, M. Henkemeyer, L.-Z. Xiong, and X.-J. Song, "Blocking EphB1 Receptor Forward Signaling in Spinal Cord Relieves Bone Cancer Pain and Rescues Analgesic Effect of Morphine Treatment in RodentsEphB1 Receptor Is Critical to Bone Cancer Pain," *Cancer research*, vol. 71, no. 13, pp. 4392–4402, 2011.

[229] Y. Drabsch and P. Ten Dijke, "TGF-β signalling and its role in cancer progression and metastasis," *Cancer and Metastasis Reviews*, vol. 31, pp. 553–568, 2012.

[230] M. S. Almén, K. J. Nordström, R. Fredriksson, and H. B. Schiöth, "Mapping the human membrane proteome: a majority of the human membrane proteins can be classified according to function and evolutionary origin," *BMC biology*, vol. 7, no. 1, pp. 1–14, 2009.

[231] S. El-Gebali, J. Mistry, A. Bateman, *et al.*, "The Pfam protein families database in 2019," *Nucleic Acids Research*, vol. 47, no. D1, pp. D427–D432, 2019. DOI: 10.1093/nar/gky995.

[232] J.-P. Himanen, M. Henkemeyer, and D. B. Nikolov, "Crystal structure of the ligand-binding domain of the receptor tyrosine kinase EphB2," *Nature*, vol. 396, no. 6710, pp. 486–491, 1998.

[233] B. Berger, M. S. Waterman, and Y. W. Yu, "Levenshtein distance, sequence comparison and biological database search," *IEEE transactions on information theory*, vol. 67, no. 6, pp. 3287–3294, 2020.

[234] B. Ramsundar, "Molecular machine learning with DeepChem," Ph.D. dissertation, Stanford University, 2018.

[235] B. Hie, E. D. Zhong, B. Berger, and B. Bryson, "Learning the language of viral evolution and escape," *Science*, vol. 371, no. 6526, pp. 284–288, 2021.

[236] M. Littmann, M. Heinzinger, C. Dallago, K. Weissenow, and B. Rost, "Protein embeddings and deep learning predict binding residues for various ligand classes," *Scientific Reports*, vol. 11, no. 1, pp. 1–15, 2021.

[237] D. Russel, K. Lasker, B. Webb, J. Velázquez-Muriel, E. Tjioe, D. Schneidman-Duhovny, B. Peterson, and A. Sali, "Putting the pieces together: integrative modeling platform software for structure determination of macromolecular assemblies," *PLoS biology*, vol. 10, no. 1, e1001244, 2012.

[238] J. Skolnick and H. Zhou, "Implications of the Essential Role of Small Molecule Ligand Binding Pockets in Protein–Protein Interactions," *The Journal of Physical Chemistry B*, vol. 126, no. 36, pp. 6853–6867, 2022.

[239] B. L. Hie, K. K. Yang, and P. S. Kim, "Evolutionary velocity with protein language models," *bioRxiv*, 2021.

[240] C. Hsu, H. Nisonoff, C. Fannjiang, and J. Listgarten, "Combining evolutionary and assay-labelled data for protein fitness prediction," *bioRxiv*, 2021.

[241] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," in *International Conference on Machine Learning*, PMLR, 2018, pp. 2323–2332.

[242] W. Jin, R. Barzilay, and T. Jaakkola, "Hierarchical generation of molecular graphs using structural motifs," in *International Conference on Machine Learning*, PMLR, 2020, pp. 4839–4848.

[243] O. J. Wouters, M. McKee, and J. Luyten, "Estimated research and development investment needed to bring a new medicine to market, 2009-2018," *Jama*, vol. 323, no. 9, pp. 844–853, 2020.

[244] G. A. Van Norman, "Drugs, devices, and the FDA: part 1: an overview of approval processes for drugs," *JACC: Basic to Translational Science*, vol. 1, no. 3, pp. 170–179, 2016.

[245] V. T. Sabe, T. Ntombela, L. A. Jhamba, G. E. Maguire, T. Govender, T. Naicker, and H. G. Kruger, "Current trends in computer aided drug design and a highlight of drugs discovered via computational techniques: A review," *European Journal of Medicinal Chemistry*, vol. 224, p. 113 705, 2021.

[246] E. Nijkamp, J. A. Ruffolo, E. N. Weinstein, N. Naik, and A. Madani, "ProGen2: exploring the boundaries of protein language models," *Cell Systems*, 2022.

[247] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *Journal of Machine Learning Research*, vol. 23, no. 120, pp. 1–39, 2022.

[248] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *arXiv preprint arXiv:2104.08691*, 2021.

[249] S. Sivarajkumar, M. Kelley, A. Samolyk-Mazzanti, S. Visweswaran, and Y. Wang, "An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing," *arXiv preprint arXiv:2309.08008*, 2023.

[250] A. Madani, B. Krause, E. R. Greene, *et al.*, "Large language models generate functional protein sequences across diverse families," *Nature Biotechnology*, vol. 41, no. 8, pp. 1099–1106, 2023.

[251] N. Ding, Y. Qin, G. Yang, *et al.*, "Parameter-efficient fine-tuning of large-scale pre-trained language models," *Nature Machine Intelligence*, vol. 5, no. 3, pp. 220–235, 2023.

[252] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[253] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," in *International Conference on Machine Learning*, PMLR, 2019, pp. 2790–2799.

[254] A. Gaber and M. Pavšič, "Modeling and structure determination of homo-oligomeric proteins: An overview of challenges and current approaches," *International journal of molecular sciences*, vol. 22, no. 16, p. 9081, 2021.

[255] S. Sledzieski, R. Singh, L. Cowen, and B. Berger, "D-SCRIPT translates genome to phenome with sequence-based, structure-aware, genome-scale predictions of protein-protein interactions," *Cell Systems*, vol. 12, no. 10, pp. 969–982, 2021.

[256] J. Szymborski and A. Emad, "RAPPPID: towards generalizable protein interaction prediction with AWD-LSTM twin networks," *Bioinformatics*, vol. 38, no. 16, pp. 3958–3967, 2022.

[257] J. Bernett, D. B. Blumenthal, and M. List, "Cracking the black box of deep sequence-based protein–protein interaction prediction," *Briefings in Bioinformatics*, vol. 25, no. 2, bbae076, 2024.

[258] E. C. Alley, G. Khimulya, S. Biswas, M. AlQuraishi, and G. M. Church, "Unified rational protein engineering with sequence-based deep representation learning," *Nature Methods*, vol. 16, no. 12, pp. 1315–1322, 2019.

[259] J. Ross, B. Belgodere, V. Chenthamarakshan, I. Padhi, Y. Mroueh, and P. Das, "Large-Scale Chemical Language Representations Capture Molecular Structure and Properties," *arXiv preprint arXiv:2106.09553*, 2021.

[260] Y. Fang, X. Liang, N. Zhang, K. Liu, R. Huang, Z. Chen, X. Fan, and H. Chen, "Mol-Instructions: A Large-Scale Biomolecular Instruction Dataset for Large Language Models," *arXiv preprint arXiv:2306.08018*, 2023.

[261] S. Chithrananda, G. Grand, and B. Ramsundar, "ChemBERTa: large-scale self-supervised pretraining for molecular property prediction," *arXiv preprint arXiv:2010.09885*, 2020.

[262] H. Cui, C. Wang, H. Maan, K. Pang, F. Luo, and B. Wang, "scGPT: Towards Building a Foundation Model for Single-Cell Multi-omics Using Generative AI," *bioRxiv*, pp. 2023–04, 2023.

[263] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl, *et al.*, "Large language models encode clinical knowledge," *Nature*, pp. 1–9, 2023.

[264] E. D. Levy, J. B. Pereira-Leal, C. Chothia, and S. A. Teichmann, "3DComplex: a structural classification of protein complexes," *PLoS Computational Biology*, vol. 2, no. 11, e155, 2006.

[265] H. Schweke, T. Levin, M. Pacesa, C. A. Goverde, P. Kumar, Y. Duhoo, L. J. Dornfeld, B. Dubreuil, S. Georgeon, S. Ovchinnikov, *et al.*, "An atlas of protein homo-oligomerization across domains of life," *bioRxiv*, pp. 2023–06, 2023.

[266] O. Avraham, T. Tsaban, Z. Ben-Aharon, L. Tsaban, and O. Schueler-Furman, "Protein language models can capture protein quaternary state," *bioRxiv*, pp. 2023–03, 2023.

[267] Y. Park and E. M. Marcotte, "Flaws in evaluation schemes for pair-input computational predictions," *Nature Methods*, vol. 9, no. 12, pp. 1134–1136, 2012.

[268] 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*, vol. 28, no. 1, pp. 235–242, 2000.

[269] N. NaderiAlizadeh and R. Singh, "Aggregating residue-level protein language model embeddings with optimal transport," *bioRxiv*, pp. 2024–01, 2024.

[270] F.-Z. Li, A. P. Amini, Y. Yue, K. K. Yang, and A. X. Lu, "Feature reuse and scaling: Understanding transfer learning with protein language models," *bioRxiv*, pp. 2024–02, 2024.

[271] A. Aghajanyan, L. Zettlemoyer, and S. Gupta, "Intrinsic dimensionality explains the effectiveness of language model fine-tuning," *arXiv preprint arXiv:2012.13255*, 2020.

[272] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms (2023)," *arXiv preprint arXiv:2305.14314*, 2023.

[273] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. hew Tang, A. Howard, H. Adam, and D. Kalenichenko, "Antization and training of neural networks for e cient integer-arithmetic-only inference," *arXiv preprint arXiv:1712.05877*, 2017.

[274] J. Liu, G. Xiao, K. Li, J. D. Lee, S. Han, T. Dao, and T. Cai, "Bitdelta: Your fine-tune may only be worth one bit," *arXiv preprint arXiv:2402.10193*, 2024.

[275] E. Nguyen, M. Poli, M. G. Durrant, *et al.*, "Sequence modeling and design from molecular to genome scale with evo," *bioRxiv*, 2024. DOI: 10.1101/2024.02.27.582234. [Online]. Available: https://www.biorxiv.org/content/early/2024/02/27/2024.02.27.582234.

[276] G. Munsamy, T. Bohnuud, and P. Lorenz, "Improving alphafold2 performance with a global metagenomic & biological data supply chain," *bioRxiv*, pp. 2024–03, 2024.

[277] C. Gorgulla, A. Nigam, M. Koop, *et al.*, *VirtualFlow 2.0 - The Next Generation Drug Discovery Platform Enabling Adaptive Screens of 69 Billion Molecules*, Apr. 2023. DOI: 10.1101/2023.04.25.537981.

[278] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in nlp," *arXiv preprint arXiv:1906.02243*, 2019.

[279] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, PMLR, 2021, pp. 8748–8763.

[280] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," in *International Conference on Machine Learning*, Pmlr, 2021, pp. 8821–8831.

[281] D. Chen, P. Hartout, P. Pellizzoni, C. Oliver, and K. Borgwardt, *Endowing Protein Language Models with Structural Knowledge*, Jan. 2024. eprint: 2401.14819 (cs, q-bio).

[282] L. Robinson, T. Atkinson, L. Copoiu, P. Bordes, T. Pierrot, and T. D. Barrett, *Contrasting Sequence with Structure: Pre-training Graph Representations with PLMs*, Preprint, Dec. 2023. DOI: 10.1101/2023.12.01.569611.

[283] A. Morehead, J. Ruffolo, A. Bhatnagar, and A. Madani, *Towards Joint Sequence-Structure Generation of Nucleic Acid and Protein Complexes with SE(3)-Discrete Diffusion*, Dec. 2023. DOI: 10.48550/arXiv.2401.06151. eprint: 2401.06151 (cs, q-bio).

[284] T. Xiao, C. Cui, H. Zhu, and V. G. Honavar, *MolBind: Multimodal Alignment of Language, Molecules, and Proteins*, Mar. 2024. eprint: 2403.08167 (cs, q-bio).

[285] B. Gao, B. Qiang, H. Tan, M. Ren, Y. Jia, M. Lu, J. Liu, W. Ma, and Y. Lan, *DrugCLIP: Contrastive Protein-Molecule Representation Learning for Virtual Screening*, Oct. 2023. eprint: 2310.06367 (cs).

[286] P. A. Fields, Y. Dong, X. Meng, and G. N. Somero, "Adaptations of protein structure and function to temperature: There is more than one way to 'skin a cat'," *The Journal of Experimental Biology*, vol. 218, no. 12, pp. 1801–1811, 2015.

[287] J. Zhao, Y. Kong, F. Zhang, and R. J. Linhardt, "Impact of temperature on heparin and protein interactions," *Biochemistry & physiology*, vol. 7, no. 2, 2018.

[288] A. Meller, M. Ward, J. Borowsky, J. Lotthammer, M. Kshirsagar, F. Oviedo, J. Ferres, and G. Bowman, "Predicting the locations of cryptic pockets from single protein structures using the pocketminer graph neural network. biorxiv 2022, 2022.06. 28.497399," *URL https://www. biorxiv. org/content/early/2022/06/29/2022.06*, vol. 28, no. 497399.1,

[289] N. Li and A. L. Girard, "Impact of ph and temperature on whey protein-proanthocyanidin interactions and foaming properties," *Food Hydrocolloids*, vol. 134, p. 108 100, 2023.

[290] A. G. de Brevern and J. Rebehmed, "Current status of ptms structural databases: Applications, limitations and prospects," *Amino acids*, vol. 54, no. 4, pp. 575–590, 2022.

[291] B. Liang, Y. Zhu, W. Shi, C. Ni, B. Tan, and S. Tang, "Sars-cov-2 spike protein post-translational modification landscape and its impact on protein structure and function via computational prediction," *Research*, vol. 6, p. 0078, 2023.

[292] Z. Peng, B. Schussheim, and P. Chatterjee, *PTM-Mamba: A PTM-Aware Protein Language Model with Bidirectional Gated Mamba Blocks*, Feb. 2024. DOI: 10.1101/2024.02.28.581983.

[293] E. D. Zhong, T. Bepler, B. Berger, and J. H. Davis, "CryoDRGN: Reconstruction of heterogeneous cryo-EM structures using neural networks," *Nature Methods*, vol. 18, no. 2, pp. 176–185, Feb. 2021, ISSN: 1548-7105. DOI: 10.1038/s41592-020-01049-4.

[294] H. Wilson and Q. Wang, "ABEL-FRET: Tether-free single-molecule FRET with hydrodynamic profiling," *Nature Methods*, vol. 18, no. 7, pp. 816–820, Jul. 2021, ISSN: 1548-7105. DOI: 10.1038/s41592-021-01173-9.

[295] G. Agam, C. Gebhardt, M. Popara, *et al.*, "Reliability and accuracy of single-molecule FRET studies for characterization of structural dynamics and distances in proteins," *Nature Methods*, vol. 20, no. 4, pp. 523–535, Apr. 2023, ISSN: 1548-7105. DOI: 10.1038/s41592-023-01807-0.

[296] L. F. Kinman, B. M. Powell, E. D. Zhong, B. Berger, and J. H. Davis, "Uncovering structural ensembles from single-particle cryo-EM data using cryoDRGN," *Nature Protocols*, vol. 18, no. 2, pp. 319–339, Feb. 2023, ISSN: 1750-2799. DOI: 10.1038/s41596-022-00763-x.

[297] B. Jing, B. Berger, and T. Jaakkola, *AlphaFold Meets Flow Matching for Generating Protein Ensembles*, Feb. 2024. DOI: 10.48550/arXiv.2402.04845. eprint: 2402.04845 (cs, q-bio).

[298] V. Lombard, S. Grudinin, and E. Laine, *Explaining Conformational Diversity in Protein Families through Molecular Motions*, Feb. 2024. DOI: 10.1101/2024.02.06.578951.

[299] B. Buntz, *Genentech's AI 'lab-in-the-loop' pioneers data-driven drug discovery*, https://www.drugdiscoverytrends.com/genentech-ai-lab-in-the-loop-drug-discovery/, Mar. 2024.

[300] M. Mirdita, K. Schütze, Y. Moriwaki, L. Heo, S. Ovchinnikov, and M. Steinegger, "Colabfold: Making protein folding accessible to all," *Nature methods*, vol. 19, no. 6, pp. 679–682, 2022.

[301] E. Bisong and E. Bisong, "Google colaboratory," *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners*, pp. 59–64, 2019.

[302] A. M. Conard, A. DenAdel, and L. Crawford, "A spectrum of explainable and interpretable machine learning approaches for genomic studies," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 15, no. 5, e1617, 2023.

[303] J. B. Kruskal, "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem," *Proceedings of the American Mathematical Society*, vol. 7, no. 1, pp. 48–50, 1956, ISSN: 00029939, 10886826.

[304] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *International Conference on Machine Learning*, PMLR, 2017, pp. 3145–3153.

[305] T. (https://math.stackexchange.com/users/85024/tad), *Measure of how much diagonal a matrix is*, Mathematics Stack Exchange, URL:https://math.stackexchange.com/q/1393907 (version: 2015-08-12). eprint: https://math.stackexchange.com/q/1393907. [Online]. Available: https://math.stackexchange.com/q/1393907.

Appendix A

Supplementary Material for Chapter 3

A.1 The corpus of experimental PPI data is limited

We sourced the PPI data in Figure 3.1 from BioGRID. While we have sourced PPI data from the STRING database everywhere else in this paper, here we chose to use BioGRID because the publication date of a PPI is easily accessible in BioGRID, allowing us to estimate the number of PPIs assayed in the last five years. While the BioGRID selection may not precisely match the STRING selection due to curation differences between the two databases, our primary aim here is conveying the relative data availability across species; this estimate should not be significantly impacted by differences in curation.

A.2 Comparison of model complexities of D-SCRIPT and PIPR

The version of PIPR that we compare to has 72,500 trainable parameters. The full D-SCRIPT model has 629,207 trainable parameters, but the vast majority of those (616,600) are in the projection module, which is a simple linear combination of all concatenated hidden states of the Bepler & Berger language model (which are themselves already redundant). The

Reagent or Resource	**Source**	**Identifier**
Deposited Data		
Protein sequence data	Szklarczyk et al., 2019	https://string-db.org/
Protein interaction data	Szklarczyk et al., 2019	https://string-db.org/
Docking benchmark dataset	Hwang et al., 2010	https://zlab.umassmed.edu/benchmark/
SARS-CoV-2 sequences and interactions	Gordon et al., 2020	PXD018117
ARS-UCD1.2 *Bos taurus* genome assembly	NCBI Assembly	GCF_002263795.1
GSE160028 *Bos taurus* RNA-seq	NCBI GEO	GSE160028
Software and Algorithms		
D-SCRIPT	Sledzieski et al., 2021	http://dscript.csail.mit.edu/
Protein sequence embeddings	Bepler & Berger 2019	tbepler/protein-sequence-embedding-iclr2019
PIPR	Chen et al., 2019	muhaochen/seq_ppi
DeepPPI	Richoux et al., 2019	richoux-f/DeepPPI
DSD	Cao et al. 2013	https://dsd.cs.tufts.edu/
CD-HIT	Fu et al., 2012	http://weizhongli-lab.org/cd-hit/
GOGO	Zhao et al., 2018	http://dna.cs.miami.edu/GOGO/
POT Python Optimal Transport	Flamary et al., 2017	https://pythonot.github.io/

Table A.1: **Key Resources Table for D-SCRIPT**

remaining stages of the model together have 12,707 trainable parameters. Practically, we prevent over-fitting through a high rate of dropout (50%) in the Projection module, combined with a very simple but structurally informed architecture in the Contact and Interaction prediction modules. Additionally, the contact map magnitude loss acts as a regularization by requiring that the predicted contact maps be sparse. Empirically, D-SCRIPT generalizes much better than PIPR, and does not seem to overfit to the training data.

Table A.2: **Rumen Target Proteins in Cow.** Jaing et al. [81] identified *TCHHL2*, *PRD-SPRRII*, *S100-A12*, and *S100-A2* as differentially expressed in sheep rumen. We identified 12 potential homologs in cow (*Bos taurus*) using BLAST with default settings. [19] (Table A.2).

Entrez Gene	Ensemble Peptide ID
TCHHL2	ENSBTAP00000066329
TCHHL2	ENSBTAP00000060747
PRD-SPRRII	ENSBTAP00000063491
PRD-SPRRII	ENSBTAP00000065881
PRD-SPRRII	ENSBTAP00000070493
PRD-SPRRII	ENSBTAP00000070470
PRD-SPRRII	ENSBTAP00000044090
S100-A12	ENSBTAP00000056088
S100-A12	ENSBTAP00000034009
S100-A12	ENSBTAP00000008523
S100-A12	ENSBTAP00000067294
S100-A2	ENSBTAP00000000589

Appendix B

Supplementary Material for Chapter 4

B.1 Maintain minimum spanning tree while sparsifying network

Consider a graph $G = (V, E)$, which we use to generate a sparsified sub-graph $G_p = (V, E_p)$ using a parameter p, which denotes the fraction of G's edges retained in G_p. We require G_p to have same connectivity as G because graph connectivity is required for many network-based link prediction methods. To ensure that G_p is connected and all the nodes in G are included in G_p, we perform the following operations, following [42].

1. Compute a random spanning tree ($T = (V, E_T)$) from G, where $|E_T| = |V| - 1$ (We used Kruskal's algorithm [303] for this, whose computational complexity is $|E| \log |V|$).

2. Let $S = E \setminus E_T$. Randomly add $p|E| - |E_T|$ edges from S to the constructed tree T to produce $G_p = (V, E_p)$.

The remaining edges in S, that were not added to construct G_p, were then used as the positive examples for experiments in Section 3.

B.2 Sparsity data set characteristics

Here we provide information about the data sets generated in the sparsified fly network analysis (Tables B.1, B.2).

Table B.1: Network Information of sparsified fly network G_p for different p-values

Sparsity	#Nodes	#Edges	Diameter	Average Degree
$p = 1.0$	3093	27134	17	17.54
$p = 0.8$	3093	21707	18	14.03
$p = 0.6$	3093	16280	17	10.52
$p = 0.4$	3093	10853	18	7.01
$p = 0.2$	3093	5426	22	3.50

Table B.2: Positive and Negative test examples for different p and k values

Sparsity	**Data Set**	**Overall**	**By shortest path bin**			
			2	3	4	5+
$p = 0.8$	Positive	5085	4824	211	36	14
	Negative	267173	8841	32942	63243	162147
$p = 0.6$	Positive	10183	9314	696	100	73
	Negative	267173	6809	26647	56987	176820
$p = 0.4$	Positive	15287	12957	1883	261	186
	Negative	267173	4652	21768	50164	190589
$p = 0.2$	Positive	20352	12343	5661	1378	970
	Negative	267173	2418	13985	35915	214855

B.3 Effect of shortest path in training network (including D-SCRIPT)

In order to illustrate the deviation in performance for predictions with different graph distances in more detail, we devise the following experiment, using graph of various sparsity:

1. For a given p, compute G_p and S_p described in Section 4.4.6.
2. For a given k, find the node-pairs in S_p having the shortest graph distance, the graph being G_p, equal to k. Call this set $R_{p,k}$.

3. Train GLIDE on the graph G_p, and compute the resulting GLIDE scores for node-pairs in $R_{p,k}$.

4. Compute scores and metrics for node-pairs in $R_{p,k}$.

This experiment was done for $k \in \{2, 3, 4, 5\}$ and $p \in \{0.8, 0.6, 0.4, 0.2\}$. Table B.3 demonstrates the corresponding AUPR scores.

One of the reasons behind the difference in AUPR results between GLIDE and D-SCRIPT/Topsy-Turvy is the significant advantage GLIDE has in predicting links between node-pairs that are very close to each other in the PPI network. As Topsy-Turvy and D-SCRIPT are not trained on any network characteristics specific to the target organism, we observe it lagging behind GLIDE in overall performance (Table B.3). However, there do appear to be regions of the network where Topsy-Turvy and D-SCRIPT perform better than GLIDE.

GLIDE is shown to be very effective in correctly predicting interaction in the core regions of the network where majority of the hub proteins and their corresponding interactions reside. On the other hand, it is far more challenging to predict interactions between proteins in the peripheral region of the PPI network, where the interactions are largely unexplored. To see if sequence-based methods like D-SCRIPT and Topsy-Turvy perform better in this region, we construct the following experiment:

1. Generate a set of hub nodes H from the complete network G, by selecting G's nodes having degree above a certain cutoff d_c.

2. Given p, construct G_p and S_p as above. Train GLIDE on G_p.

3. Filter out protein-pairs from S_p if either of the protein is contained in H to produce S'_p.

4. For a given k, find the protein-pairs in S'_p having the shortest graph distance, the graph being G_p, equal to k. Call this set $R_{p,k}$.

5. Compute scores and metrics fo rthe pairs in $R_{p,k}$.

We report AUPR scores for D-SCRIPT, GLIDE and Topsy-Turvy on the hub-free data sets in Table B.4.

Table B.3: AUPR scores for D-SCRIPT, Topsy-Turvy, and GLIDE, for different values of k and p, including hub nodes

Sparsity	**Model**	**Overall AUPR**	**AUPR by Shortest Path**			
			2	3	4	5+
$p = 0.8$	GLIDE	**0.8057**	**0.8370**	**0.1186**	0.0016	0.0003
	D-SCRIPT	0.1256	0.5169	0.0184	**0.0039**	**0.0007**
	Topsy-Turvy	0.2442	0.5506	0.0260	0.0007	0.0002
$p = 0.6$	GLIDE	**0.8398**	**0.8847**	**0.1379**	0.0041	0.0009
	D-SCRIPT	0.2051	0.7159	0.0583	0.0060	0.0009
	Topsy-Turvy	0.3668	0.7412	0.0781	**0.0072**	**0.0019**
$p = 0.4$	GLIDE	**0.8180**	**0.8763**	**0.2612**	0.0111	0.0035
	D-SCRIPT	0.2762	0.8376	0.1385	0.0234	0.0037
	Topsy-Turvy	0.4529	0.8529	0.1846	**0.0253**	**0.0112**
$p = 0.2$	GLIDE	**0.7379**	0.8256	**0.6702**	**0.1337**	0.0112
	D-SCRIPT	0.3277	0.9161	0.4836	0.0734	0.0123
	Topsy-Turvy	0.5095	**0.9224**	0.5430	0.1171	**0.0311**

Table B.4: AUPR scores for D-SCRIPT, Topsy-Turvy, and GLIDE, for different values of k and p, after the removal of hub nodes.

Sparsity	**Model**	**Overall AUPR**	**AUPR by Shortest Path**			
			2	3	4	5+
$p = 0.8$	GLIDE	**0.3993**	**0.4857**	0.1094	0.0022	0.0004
	D-SCRIPT	0.0143	0.2509	0.0365	0.0082	**0.0015**
	Topsy-Turvy	0.0389	0.3141	**0.0545**	**0.0375**	0.0004
$p = 0.6$	GLIDE	**0.4535**	**0.5910**	0.1336	0.0054	0.0011
	D-SCRIPT	0.0280	0.4436	0.1035	0.0114	0.0015
	Topsy-Turvy	0.0804	0.5422	**0.1528**	**0.0142**	**0.0031**
$p = 0.4$	GLIDE	**0.4329**	0.6055	0.2022	0.0150	0.0037
	D-SCRIPT	0.0398	0.5624	0.1760	**0.0395**	0.0063
	Topsy-Turvy	0.1067	**0.6562**	**0.2245**	0.0150	**0.0192**
$p = 0.2$	GLIDE	**0.3274**	0.5956	0.3068	0.0822	0.0094
	D-SCRIPT	0.0521	0.6112	0.3033	0.0910	0.0175
	Topsy-Turvy	0.1359	**0.6970**	**0.3986**	**0.1539**	**0.0399**

Appendix C

Supplementary Material for Chapter 5

C.1 DUD-E train/test splits

Targets were randomly split within each class, so that there were examples of each class in the training and test set. There are a total of 57 targets, 26 in training and 31 in testing. A list of all targets and classes evaluated in train and test are in Table C.1.

C.2 Choice of protein and molecule featurization

C.2.1 Alternate options for target feature generation

We explored several choices for using language models to generate protein features, including D-SCRIPT [11], Prose [48], [53], ESM [208], and ProtBert [209]. For D-SCRIPT, we use the 100 dimensional embeddings after the first projection layer. For the other language models, we use the output of the final embedding layer. All models provide per-amino acid features, which we average along the length of the protein to get a fixed length vector. Table C.2 shows the performance of a ConPLex model using each of the featurizations (without contrastive learning for simplicity). While there is no one language model which is uniformly best (Section 5.5), we chose to use ProtBert which seemed to have consistent strong performance.

Table C.1: **DUD-E target classes and splits.**

Target ID	Class	Split
AA2AR	Gpcr	Train
ABL1	Kinase	Train
AKT1	Kinase	Train
CDK2	Kinase	Train
EGFR	Kinase	Train
FAK1	Kinase	Train
FGFR1	Kinase	Train
IGF1R	Kinase	Train
JAK2	Kinase	Train
KIT	Kinase	Train
MAPK2	Kinase	Train
MK01	Kinase	Train
PLK1	Kinase	Train
TGFR1	Kinase	Train
ANDR	Nuclear	Train
ESR2	Nuclear	Train
MCR	Nuclear	Train
PPARD	Nuclear	Train
THB	Nuclear	Train
ACE	Protease	Train
BACE1	Protease	Train
DPP4	Protease	Train
FA7	Protease	Train
HIVPR	Protease	Train
MMP13	Protease	Train
TRYB1	Protease	Train
ADRB1	Gpcr	Test
ADRB2	Gpcr	Test
CXCR4	Gpcr	Test
DRD3	Gpcr	Test
AKT2	Kinase	Test
BRAF	Kinase	Test
CSF1R	Kinase	Test
KPCB	Kinase	Test
LCK	Kinase	Test
MET	Kinase	Test
MK10	Kinase	Test
MK14	Kinase	Test
MP2K1	Kinase	Test
ROCK1	Kinase	Test
SRC	Kinase	Test
VGFR2	Kinase	Test
WEE1	Kinase	Test
ESR1	Nuclear	Test
GCR	Nuclear	Test
PPARA	Nuclear	Test
PPARG	Nuclear	Test
PRGR	Nuclear	Test
RXRA	Nuclear	Test
ADA17	Protease	Test
CASP3	Protease	Test
FA10	Protease	Test
LKHA4	Protease	Test
RENI	Protease	Test
THRB	Protease	Test
TRY1	Protease	Test
UROK	Protease	Test

C.2.2 Alternate options for drug feature generation

We also explored three different choices for generating drug features, including the Morgan fingerprint [210], MolR [213], and Mol2Vec [212]. Results are shown in Table C.3. The Morgan fingerprint uniformly outperforms the other choices, and so we chose to use it for ConPLex.

Table C.2: **ConPLex classification results on benchmark data sets using different PLMs to generate protein features.**

Benchmark	Protein Features	AUPR	AUROC	F1
BIOSNAP	D-SCRIPT	0.911	0.897	0.831
	Prose	**0.914**	**0.898**	**0.838**
	ESM	0.898	0.876	0.817
	ProtBert	0.895	0.873	0.811
BindingDB	D-SCRIPT	0.618	0.870	0.619
	Prose	0.618	0.862	0.625
	ESM	0.637	**0.881**	**0.637**
	ProtBert	**0.652**	0.876	0.636
DAVIS	D-SCRIPT	0.475	0.912	0.533
	Prose	0.463	0.907	0.523
	ESM	0.480	0.916	0.544
	ProtBert	**0.511**	**0.917**	**0.546**
BIOSNAP Unseen Proteins	Prose	**0.875**	**0.868**	**0.798**
	ESM	0.850	0.839	0.770
	ProtBert	0.841	0.827	0.750
BIOSNAP Unseen Drugs	Prose	**0.881**	**0.857**	**0.802**
	ESM	0.876	0.850	0.796
	ProtBert	0.876	0.848	0.785

Table C.3: **ConPLex classification results on benchmark data sets using different methods to generate small molecule features.**

Benchmark	Drug Features	AUPR
BIOSNAP	Morgan	**0.904**
	MolR	0.853
	Mol2Vec	0.792
BindingDB	Morgan	**0.668**
	MolR	0.578
	Mol2Vec	0.380
DAVIS	Morgan	**0.511**
	MolR	0.436
	Mol2Vec	0.144

C.2.3 Feature attribution reveals information gain from tuning on PPI

We additionally investigated training ConPLex models with features generated by concatenating protein language model embeddings with embeddings from a D-SCRIPT [11] model pre-trained on human PPIs. We refer to the output of the first projection module of D-SCRIPT, which takes as input the $n \times d$ protein representation and tunes it to an $n \times 100$ representation. We then concatenate this with the original PLM representation to get an embedding with $d + 100$ features.

While the top-line performance of the augmented models are similar to the base models, an attribution study using DeepLift [304] shows that the new D-SCRIPT-derived features are disproportionately represented in the set of highly important features (Figure C.1). This suggests that tuning on a related task refines the representations from the general protein language models to ones more suited for the specific task, as in [188]. This explanation is supported by the fact that the 100-dimensional D-SCRIPT features alone achieved only slightly decreased performance on the DTI task compared to PLM-based models with 10-50x as many parameters (Table C.2).

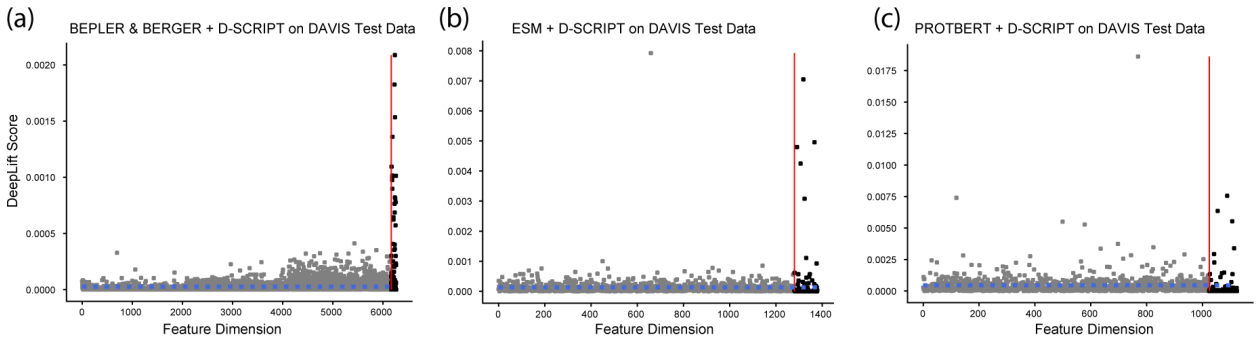


Figure C.1: DeepLift feature attributions for Prose **(a)**, ESM **(b)**, and ProtBert **(c)** embedding dimensions (gray) and the respective D-SCRIPT embedding dimensions (black). D-SCRIPT features have an outsize contribution to the overall model prediction relative to the PLM features alone. The dashed blue line indicates the mean feature attribution.

C.3 Training and inference time analysis

One of the benefits of using pre-trained models for feature generation is that computation times are amortized over the lifespan of downstream applications. Pre-trained models incur an up-front computational cost, but can then be re-used for multiple inference tasks with straightforward architectures. Our framework allows for training DTI models up to 8 times faster than an end-to-end method like MolTrans [185]. Inference of DTIs is also roughly an order of magnitude faster (Table C.4). MolTrans has a faster per-epoch training time than ConPLex on DAVIS, but a slower wall clock time and much slower inference.

Table C.4: Comparison of training and inference times on three high-coverage data sets with contrastive training on DUD-E (mean seconds over 5 runs).

Benchmark	**Model**	**Wall Clock**	**Training (per epoch)**	**Inference**
BIOSNAP	ConPLex	**1273**	**22.84**	**0.556**
	MolTrans	6424	116.21	31.64
BindingDB	ConPLex	**1302**	**24.21**	**2.168**
	MolTrans	9874	142.96	222.17
DAVIS	ConPLex	**1145**	22.38	**0.619**
	MolTrans	1417	**15.38**	35.06

Appendix D

Supplementary Material for Chapter 6

D.1 Protein-Protein Interaction Prediction

D.1.1 Learning rate selection for fine-tuning

Due to the substantially larger number of parameters, fine-tuned models are more prone to over-fit to the training data. As such, it is common to use lower learning rates when performing full fine-tuning compared to training only the classification head (MLP) or a small number of added paramters (PEFT). In Table D.1, we show the result of fine-tuning with several different learning rates. The results presented in the main text (Table 6.2) uses a learning rate of $5e-4$.

D.1.2 Adaptation of increasingly deeper layers yields improved model performance

One choice when selecting designing a parameter-efficient fine-tuning setup for a protein language model is which and how many transformer layers to adapt. We show in Table D.2 that, on the PPI prediction task, increasing numbers of adapted layers (with more parameters) leads to generally stronger performance (although this is not a uniform increase

Table D.1: Fine-tuning a larger number of parameters typically requires a lower learning rate to reduce over-fitting. We experiment with 5 different learning rates from $1e-3$ to $1e-5$, finding the best performance at a learning rate of $5e-4$, half of what is used for MLP and PEFT models.

Learning Rate	Val. AUPR	AUPR	Acc.	F1	MCC	Prec.	Rec.	Spec.
1e-3	0.605	0.576	0.548	0.568	0.097	0.544	0.594	0.502
5e-4	**0.622**	**0.623**	**0.604**	0.631	**0.210**	0.591	**0.676**	0.532
1e-4	0.615	0.604	0.603	0.622	0.207	**0.594**	0.653	0.553
5e-5	0.619	0.596	0.562	0.480	0.130	0.590	0.405	**0.718**
1e-5	0.532	0.536	0.500	**0.667**	0.000	0.500	1.000	0.000

due to variance in model training). However, it is not immediately clear that the best strategy is to adapt subsequent layers from the end. Although this is theoretically sound, we compare the following alternative approaches on the symmetry prediction task (test set AUPR in parenthesis): adapting the last 5 layers (0.400), adapting the first 5 layers (0.215), adapting 5 intermediate layers (layers 14-19; 0.344), adapting 5 randomly selected layers (layers 3, 6, 12, 17, 23; 0.332). We find significantly degraded performance with all options except for the last 5 layers. Performance is especially bad when adapting only the first 5 layers. This corresponds with the generally-understood principle that early layers of a protein language model learn broad abstractions of the data, while later layers learn task-specific features.

Table D.2: We find that model performance generally increases as increasingly many layers are adapted from the end of the model.

# Layers	Val. AUPR	AUPR	Acc.	F1	MCC	Prec.	Rec.	Spec.
1	0.612	0.605	0.524	0.586	0.171	0.586	0.586	**0.586**
2	0.615	0.622	0.602	0.632	0.208	0.589	0.681	0.524
4	0.619	0.597	0.588	0.630	0.180	0.572	0.701	0.475
5	**0.640**	**0.633**	0.601	0.630	0.204	0.587	0.680	0.522
6	0.627	0.624	**0.604**	**0.643**	**0.213**	0.586	**0.711**	0.497

D.1.3 PEFT and FT training and validation curves

We show training and validation loss curves, as well as validation AUPR curves, over training in Figure D.1. In Figure D.2, we show training and validation loss curves, as well as validation AUPR curves, for all different combinations of Q/K/V matrices tested in Table 6.4. In Figure D.3, we show training and validation loss curves, as well as validation AUPR curves, for LoRA rank $1, 2, 4, 8, 64$ as tested in Table 6.5.

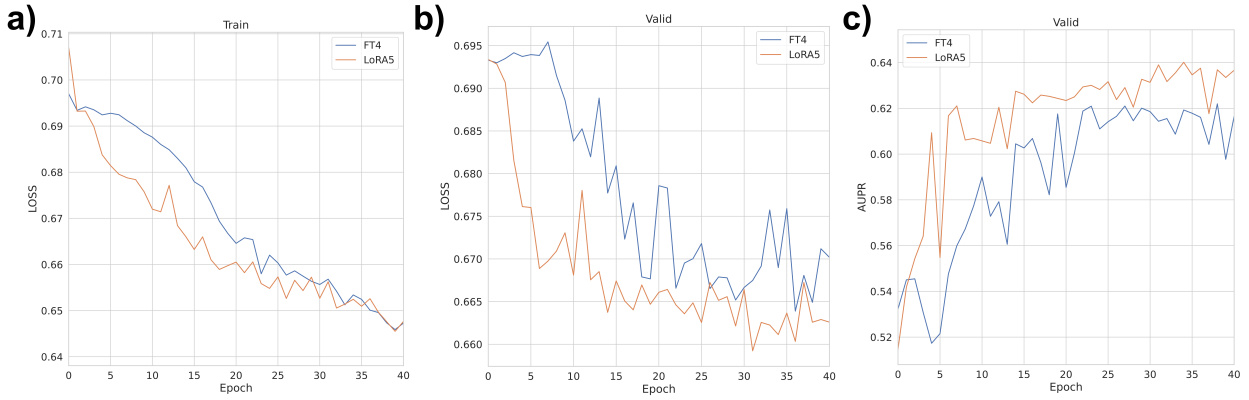


Figure D.1: Training **(a)** and validation **(b)** loss curves, AUPR curves **(c)** for FT (4 layers) and PEFT (5 layers) from Table 6.2. Note that all other training parameters were held constant between these runs.

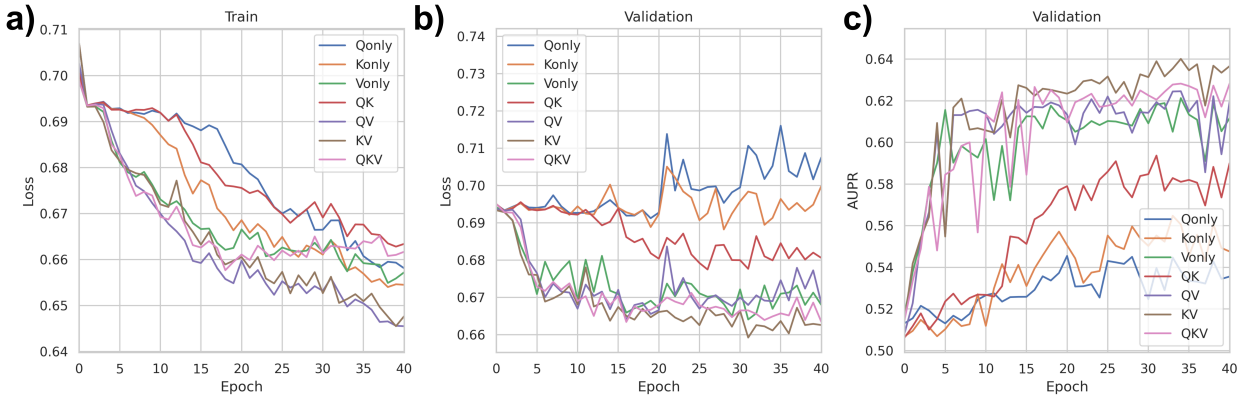


Figure D.2: Training **(a)** and validation **(b)** loss curves, AUPR **(c)** curves for models trained with LoRA adapters on Q, K, V, QK, QV, KV, QKV matrices from Table 6.4. Note that all other training parameters were held constant between these runs.

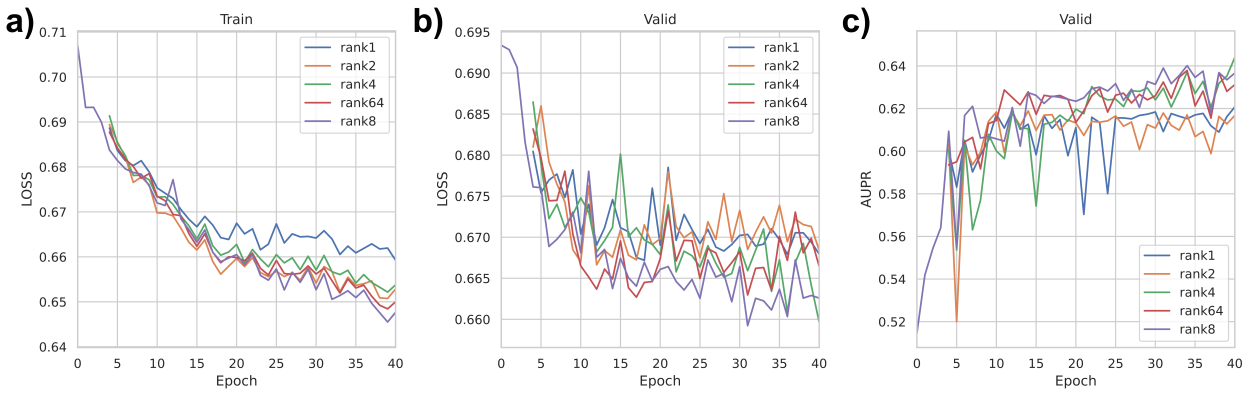


Figure D.3: Training **(a)** and validation **(b)** loss curves, AUPR **(c)** curves for models trained with LoRA ranks $r = 1, 2, 4, 8, 64$ from Table 6.5. Note that all other training parameters were held constant between these runs.

D.1.4 Baseline MLP Model

As a baseline to compare with fine-tuning, we train an MLPClassifier model from scikit-learn using embeddings extracted from ESM2 (650M parameters). Parameters for the MLPClassifier were selected by cross-validation on macro average precision over a grid search. We searched over all combinations of

- $activation = [\text{“}logistic\text{”}, \text{“}relu\text{”}, \text{“}identity\text{”}]$
- $alpha = [0.0001, 0.001, 0.01]$
- $learning\_rate\_init = [0.001, 0.01]$
- $max\_iter = 1000, 2000$
- $hidden\_layer\_sizes = [(64,), (128,), (512,), (64, 64), (128, 128), (64, 64, 64)]$
- $tol = [1e-4, 1e-5]$

In Figure D.4, we show that this MLP model is well calibrated (fraction of predicted positives at threshold p roughly matches predicted probability p).

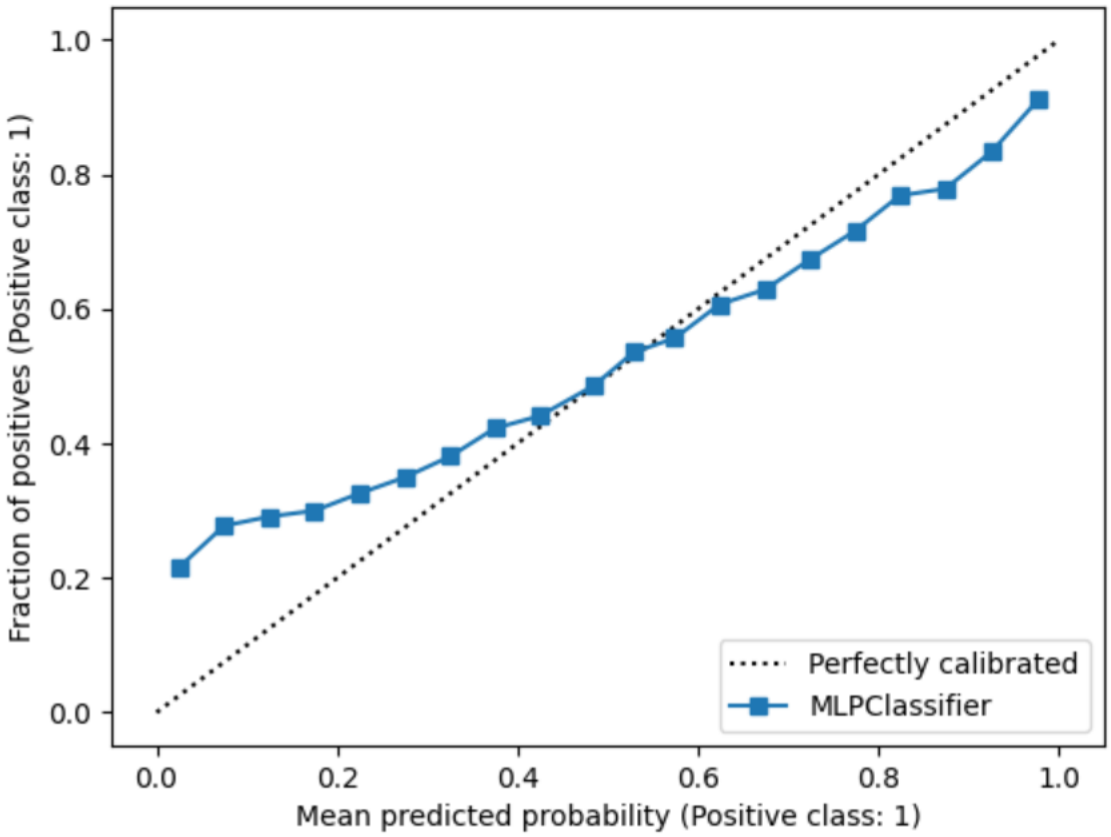


Figure D.4: Calibration curve for sklearn MLP on embeddings from a frozen ESM2 model, trained for PPI prediction on the benchmarks from Bernett et al. [257].

D.2 Homooligomer Symmetry Prediction

D.2.1 Test Set Support

Homooligomer symmetry prediction is a highly unbalanced, multi-class prediction task. In Table D.3, we show the number of examples in the test set for each class. We show in Figure 6.3 that PEFT and MLP models are competitive for high support classes like C1, C2, C5, D2, and I, while the FT model is substantially better on rare classes like C7-C9, D4, D5, and O.

Table D.3: **Test Set Support.** Number of examples of each symmetry class in the test set. This data is highly imbalanced, with most examples having either C1, C2, D2, or D3 symmetry.

Symmetry Class |	Support
C1	29,773
C2	23,393
C3	5,516
C4	3,507
C5	6,055
C6	2,920
C7-C9	1,406
C10-C17	1,910
D2	10,370
D3	8,436
D4	2,052
D5	1,800
D6-D12	2,045
H	4,014
O	540
T	1,927
I	5,232
Other	328

D.2.2 Fine-tuning all layers

We perform an additional experiment wherein we fine-tune *all* layers of the protein language model– we note that this is substantially more compute intensive and requires us to decrease

our batch size to 4. We also reduce the learning rate to 0.0005 to account for the larger number of parameters. We find that on the symmetry task, this model achieves a test set AUPR of 0.446, which is competitive with but does not exceed the performance of the models in Table 6.3. Thus, when we refer to "fine-tuning", we consider training all parameters of a fixed subset of layers in the model.

D.3 Visualizing Attention

We visualize attention values with and without parameter-efficient adapter updates for the last five transformer layers of the PEFT model trained on PPI prediction from Table 6.2. We average the output of all 20 attention heads so that for a protein of length n, we get a matrix of size $n \times n$. LoRA weights are turned on or off with the `peft` package commands `disable_adapter_layers` and `enable_adapter_layers`. We find that with fine-tuning on PPI prediction, attentions are spread out much further from the diagonal, indicating more distal attention necessary for predicting protein-protein interactions.

We compute a measurement of the extent to which attention is concentrated along the diagonal using the Pearson's sample correlation [305]. Given a protein of length N, we treat the attention from residue i to residue j in matrix $A \in \mathbb{R}^{N \times N}$ as a pair of samples (i, j) from discrete distributions X and Y, where the magnitude of the attention $A_{i,j}$ corresponds to the probability mass of this pair of indices, or the weight of each sample. Then, the total "sample size" is the total magnitude of attention $n = \sum_{i,j \in N} A_{ij}$. We can then calculate the diagonal correlation of A as

$$r_{xy} = \frac{(n \sum_{i,j \in N} A_{ij} ij) - (\sum_{i,j \in N} A_{ij} i - \sum_{i,j \in N} A_{ij} j)}{\sqrt{n \sum_{i,j \in N} A_{ij} i^2 - (\sum_{i,j \in N} A_{ij} i)^2} - \sqrt{n \sum_{i,j \in N} A_{ij} j^2 - (\sum_{i,j \in N} A_{ij} j)^2}} \tag{D.1}$$

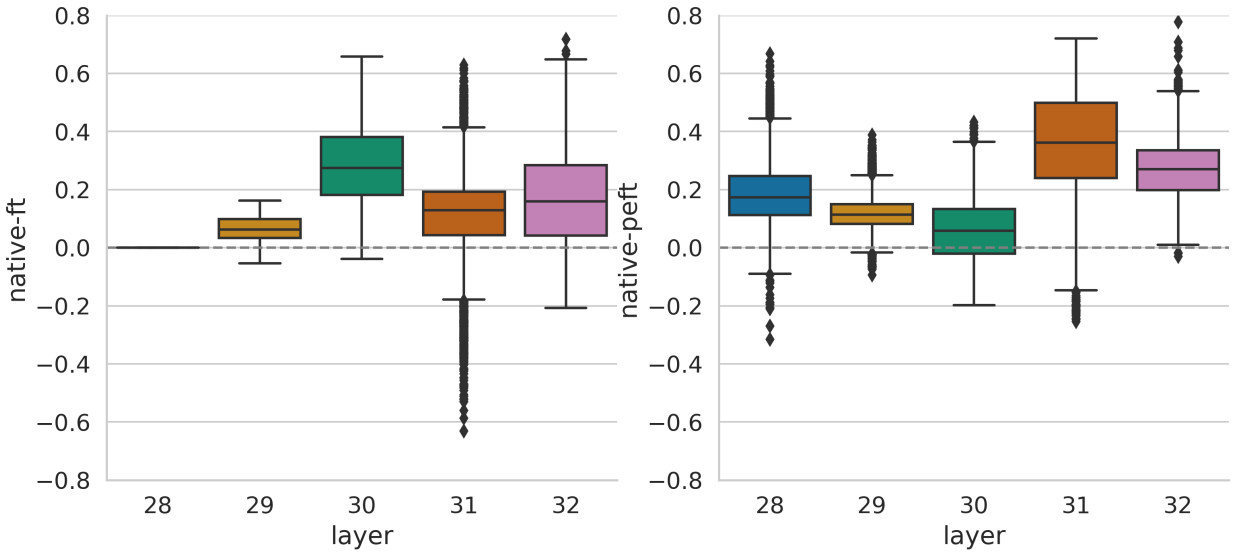


Figure D.5: We see similar increases in global attention in both traditional fine-tuning **(a)** and parameter-efficient fine-tuning **(b)**, when quantifying it with the diagonal correlation r_{xy}, described in detail in Equation D.1.