# GEOMETRIC DEEP LEARNING AND GENERATIVE MODELING OF 3D BIOMOLECULES

---

A Dissertation presented to

the Faculty of the Graduate School

at the University of Missouri

---

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

ALEX MOREHEAD

Dr. Jianlin Cheng, Dissertation Supervisor

MAY 2025

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

# GEOMETRIC DEEP LEARNING AND GENERATIVE MODELING OF 3D BIOMOLECULES

presented by Alex Morehead,

a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

---

Dr. Jianling Cheng

---

Dr. Dong Xu

---

Dr. Xiaoqin Zou

---

Dr. Derek Anderson

# DEDICATION

I dedicate this dissertation to my parents and siblings for their unwavering love and support through the merry-go-round of life.

# ACKNOWLEDGMENTS

To start, I want to deeply thank Dr. Jianlin Cheng for his caring and tireless mentorship over the last five years. I have learned a tremendous amount about the process and philosophy of doing open science through my formative years in Dr. Cheng's Bioinformatics & Machine Learning (BML) research lab. The community of creative, motivated, and talented individuals Dr. Cheng has cultivated in his lab has inspired several generations of researchers, including my own, to explore their curiosities and push their intellectual limits. I am fortunate to have received Dr. Cheng's guidance and support throughout my graduate studies.

Next, I'd like to thank my committee members, Dong Xu, Xiaoqin Zou, and Derek Anderson, for their continual support and constructive feedback throughout my Ph.D. journey. Their insights on the intricacies and implications of my research have challenged me to carefully consider the societal impacts of my research beyond the confines of publication and to see the bigger picture of academic research.

Reflecting on my scientific trajectory, I am reminded of my gratitude for the countless people along the way who transformed my academic experience from one more typical to one "seasoned with serendipity". In particular, I am grateful for the education and mentorship I received from Dr. Jeffrey Poet at Missouri Western State University as an undergraduate student in computer science. It was in Jeff's multivariate calculus course that I developed a fascination with the intersection of calculus and geometry and discovered a thrilling implication of combining time-tested mathematical theory with modern computer science: the rapid advancement of deep learning and artificial intelligence. Through Jeff's guidance during a summer undergraduate research project, I was also introduced to the nascent intersection of computational research and synthetic biology, which would foreshadow my path through science more than I could have imagined at the time.

My initial interest in artificial intelligence research was reinforced by my experience learning from and researching with Dr. Joseph Kendall-Morwick at Missouri Western State University. Through the twists and turns of life's merry-go-round, this experience ultimately led me to work with Dr. George Mohler at IUPUI during a summer data science research experience I had alongside many brilliant, driven, and kind undergraduates, including Lauren Ogden, Gabe Magee, Dwight Sablan, Theo Carr, and many others, for which I am incredibly grateful.

Soon after joining the BML lab, I met and collaborated with many amazing and resilient individuals. Starting my Ph.D. studies during a pandemic certainly presented unique challenges in connecting and developing new ideas together, but we all soon found a "new normal" we could settle into, exchanging long turns at a physical whiteboard for hours-long Zoom calls dissecting research papers and debugging neural network source code. Namely, I want to thank my lab mates in the BML lab, including Chen Chen, Xiao Chen, Jian Liu, Nabin Giri, Frimpong Boadu, Sajid Mahmud, Farhan Quadir, Tianqi Wu, Zhiye Guo, Raj Roy, Ashwin Dhakal, Elham Soltanikazemi, Max Highsmith, Rajan Gyawali, Joel Selvaraj, Akshata Hegde, Pawan Neupane, Shreya Basnet, and Tom Nguyen. From each of you, I gleaned valuable lessons and perspectives on scientific research and life more generally.

During the midpoint of my Ph.D. studies, I also had the great pleasure of serendipitously meeting and collaborating remotely with several incredible people at the University of Cambridge and beyond, including Arian Jamasb, Chaitanya Joshi, Simon Mathis, Rishabh Anand, Charles Harris, and Zuobai Zhang. I fondly recall the (multiple!) times we were locked to our laptops scurrying to prepare machine learning conference papers, submitting them with only seconds to spare before their international deadlines. These experiences taught me the value of dedication to one's craft and creative endeavors as well as the impact of perfectly timed AI memes.

Two other inflection points of good fortune during my Ph.D. came through a

virtual poster session in which I met Joshua Meier, with whom I would eventually do an internship at Absci, and an unexpected online message from Jeffrey Ruffolo, with whom I would later work as an intern at Profluent Bio. Both of these internship experiences were pivotal for my growth and development as a researcher, and I am ever grateful for the mentorship I received in each.

After arriving at the University of Missouri in Columbia, I met several people who became a core part of my local community, including Roland Oruche, Tyler Reaves, Gbenga Omotara, Jaired Collins, Nick Carson, Tyler Stanford, Yeongjun Kim, and Liam Kincaid. I am grateful for my time spent with each, whether it was a one-off excursion to a local pizza joint, ramblings on the impact of AI on scientific research, or a late-night deep dive on the implications of determinism vs. free will. As a result, we all learned from and challenged each other to expand our worldviews.

Last but certainly not least, I want to thank my parents, Barry and Cynthia Morehead, for their unrelenting support of my academic and extracurricular pursuits from an early age. Each helped me cultivate a deep sense of wonder, curiosity, and reverence of nature and those who inhabit it. Finally, my siblings and close family, Lauren, Nick, Michael, McKenzie, and Brad, also have my deepest gratitude and love. We've been through thick and thin, and I'm thankful we get to do life together.

*"The cure to boredom is curiosity. There is no cure for curiosity."*

–Dorothy Parker, Poet and Writer

*"I want to understand the big questions, the really big ones that you normally go into philosophy or physics if you're interested in. I thought building AI would be the fastest route to answer some of those questions."*

–Demis Hassabis, DeepMind

# TABLE OF CONTENTS

# LIST OF TABLES

Table Page

# LIST OF FIGURES

Figure Page

# ABSTRACT

Life's molecules, ranging from small molecule ligands to large polymer proteins, are intricately responsible for the biomolecular functions that maintain life within and beyond a single cell. Nonetheless, such biomolecules and their structural roles in cellular biology remain poorly understood at the genomic scale owing to their complex inter-atomic interactions, necessitating the development of new computational methods for studying biomolecules at the atomic level.

To address this issue, in this dissertation, I describe the development of a collection of deep learning methods (Geometric Transformers, GCPNet, GCDM, and FlowDock) for modeling increasingly complex biomolecular structures and interactions. These methods have advanced the state-of-the-art of deep learning in protein and biomolecular representation learning, generative modeling of 3D molecules, and protein-ligand structure and affinity prediction. Additionally, in this dissertation, I detail the design and results of a new deep learning benchmark (PoseBench) and ensembling prediction method (MULTICOM_ligand) for standardized and broadly applicable protein-ligand docking and structure prediction. The findings of the former benchmark suggest that future work in deep learning for 3D biomolecules may benefit from stronger dataset splitting and out-of-distribution evaluation. Further, the latter ensembling method ranked as a top-5 method in the ligand prediction category of the 16th Critical Assessment of Techniques for Structure Prediction (CASP16).

Taken together, this dissertation represents an advancement in our understanding of life's molecules through the lens of deep learning as well as new insights and directions for future deep learning research in the physical and life sciences. All methods, benchmarks, and datasets described in this dissertation have been open sourced and made freely available to the scientific community.

# Chapter 1

# INTRODUCTION

## 1.1 THE BIOMOLECULAR BASIS OF LIFE

Life is driven by collections of interacting atoms known as molecules, and the three-dimensional (3D) shape of these molecules defines their biochemical function within and beyond a single cell [1], enabling complex cellular ecosystems to develop and self-regulate [2]. The central dogma of molecular biology, visualized in Figure 1.1, succinctly characterizes the origins of these molecules [3], namely that precursor molecules known as deoxyribonucleic acids (DNA), once transcribed into messenger ribonucleic acids (RNA), store the necessary genetic information to be translated into new polymer chains of amino acids, known as proteins, within a cell's ribosome. These protein biomolecules, known as the "workhorses of the cell", perform many of the most common tasks in cellular biology [4].

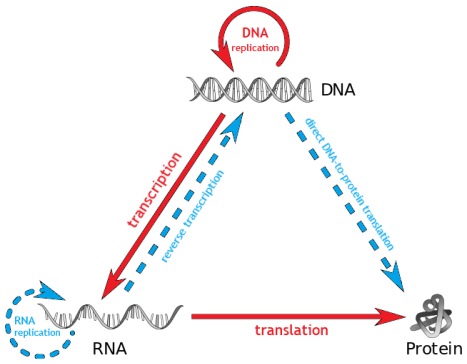



Figure 1.1: An illustration of information flow in molecular biology's central dogma.

As the 3D structure of a protein largely describes its function in living organisms [5], determining the structure of a protein according to its amino acid sequence is crucial to understanding its multi-faceted cellular roles. Over the last 50 years, the most common way scientists have set out to deduce such structures was through laborious and expensive experimental techniques such as X-ray crystallography [6], nuclear magnetic resonance spectroscopy [7], and cryo-electron microscopy [8], making it difficult to ascertain the structure of proteins beyond a small fraction of those present in nature. However, recent progress in computational prediction of 3D protein structures from their primary amino acid sequences, which are readily available in vast quantities, now allows researchers to quickly estimate protein structures at a genomic scale, ushering in a new era of computational and structural biotechnology. Such progress was largely driven by advances in artificial intelligence (AI), machine learning (ML), and particularly deep learning (DL) methodologies, suggesting that AI-driven methods in the physical sciences may provide similar levels of scientific impact in related disciplines in the coming years. To contextualize the contributions of this dissertation, in the next two sections, I will outline key paradigms in machine learning and how they relate to recent progress in our ability to model, understand, and design biomolecular systems at scale.

## 1.2 MACHINE LEARNING PARADIGMS

The field of machine learning is found within a hierarchy originating with general-purpose AI algorithms and giving rise to deep learning methods based on artificial neural networks. The relationship between these computational (sub)disciplines is depicted in Figure 1.2. As this figure shows, machine learning also enables two distinct learning paradigms, discriminative modeling and generative modeling, with the former focused on identifying optimal partitioning of a dataset for predictive or classification purposes and the latter aiming to reproduce an arbitrary data-generating

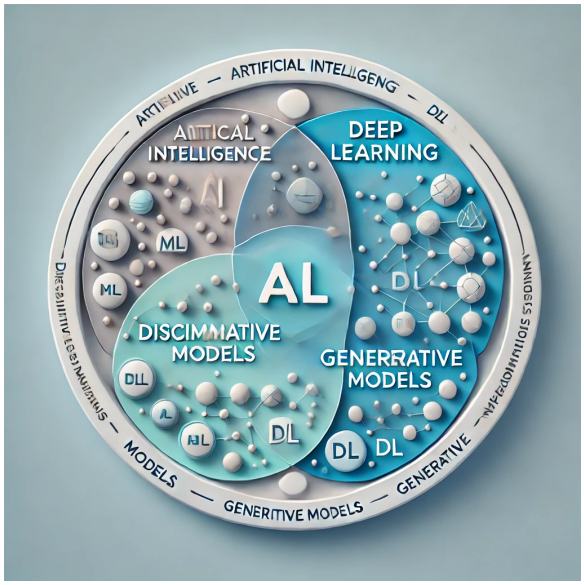



Figure 1.2: A taxonomy of artificial intelligence and machine learning methods.

process for exploratory or creative endeavors. Machine learning (and thereby deep learning) can further be divided into supervised and unsupervised learning. In broad strokes, supervised learning uses labeled data to teach a machine learning model to predict the labels of new data points, whereas unsupervised learning uses (potentially) unlabeled data to train a machine learning model to cluster or distributionally characterize a dataset to group or sample new data points [9]. Generally speaking, discriminative models are supervised learning algorithms, while generative models are often trained in an unsupervised manner.

Whereas machine learning algorithms often refer to classical learning techniques such as linear regression and decision trees, deep learning represents a dedicated class of learning algorithms based on deep neural networks trained using backpropagation and gradient descent [10]. As deep learning methods have increasingly become instrumental in modern scientific research, they will be the primary type of machine learning algorithm discussed in the remainder of this dissertation. The most common type of deep learning algorithm today is a neural network model architecture referred to as a Transformer [11]. The key components of a Transformer are based on

the idea of attention, in particular self-attention, which allows the model to implicitly learn an importance score between any pair of units in the model's input. For example, in the context of training on natural language data using self-supervised learning (a popular form of unsupervised learning based on randomly masking input units), Transformers can learn the underlying relationships between words in a sentence, relationships which give rise to complex grammatical patterns and higher-order narrative structures. Similarly, sparse variants of Transformers operating on graphs, known as graph neural networks, can learn real-valued functions of graph-structured data for node, edge, or graph-level discriminative and generative tasks [12].

## 1.3 MACHINE LEARNING FOR BIOMOLECULES

Although deep learning was initially debuted primarily in the realm of computer vision [13], applications of deep learning in the physical sciences have emerged as a fruitful independent research paradigm over the last several years [14]. AlphaFold 2 [15], one of the first foundational deep learning methods in the physical sciences, namely for protein structure prediction, garnered significant interest from the broader scientific community for its clear demonstration that, when designed for the right types of problems, deep learning algorithms can considerably accelerate scientific discovery through their powerful predictive modeling capabilities in complex data domains. This method sparked a slew of new research [16, 17, 18, 19, 20, 21, 22, 23, 24, 25] in geometric deep learning, a branch of deep learning studying the symmetries present in diverse (e.g., scientific) data sources and, for sake of learning efficiency and expressiveness, how to model them directly by designing appropriate inductive biases within a given neural network architecture [26]. Notably, AlphaFold's focus on protein-specific structure prediction was soon expanded to encompass all of life's (bio)molecules with AlphaFold 3 [27], with several precursor works [28, 29, 30, 31, 32] building in this research direction.

## 1.4 DISSERTATION OUTLINE

This dissertation presents a series of deep learning methods developed to advance our understanding of biomolecular structures and their interactions for drug discovery and design. The next four chapters describe methods for protein representation learning and interaction prediction; biomolecular representation learning; small molecule generation and optimization and structure-based drug design; and protein-ligand docking and affinity prediction, respectively. In contrast, the final two chapters detail a deep learning benchmark and an ensembling method ranked as a top-5 ligand prediction method in the CASP16 competition.

In Chapter 2, I describe the new Geometric Transformer graph neural network architecture for protein representation learning and analyze its performance for an important task in protein structural modeling: prediction of atomic protein-protein interactions. In Chapter 3, I present the Geometry-Complete Perceptron Network (GCPNet) for representation learning of 3D biomolecules and characterize its predictive performance for a range of scientific tasks in biology, chemistry, and physics. In Chapter 4, I detail the Geometry-Complete Diffusion Model (GCDM) for small molecule generation and optimization which adapts GCPNet for expressive diffusion generative modeling of 3D molecules as well as structure-based drug design. In Chapter 5, I present a new conditional flow matching method named FlowDock for protein-ligand docking and affinity prediction and contextualize its structure prediction performance using standardized benchmarking data and its affinity prediction results in the CASP16 ligand prediction competition.

In Chapter 6, I then discuss the new deep learning benchmark PoseBench for broadly applicable protein-ligand docking and structure prediction, the results of which highlight the importance in future work of rigorously evaluating the generalization capabilities of new generative structure prediction models and assessing

their ability to balance biomolecular structure prediction and protein-ligand interaction modeling accuracy. In Chapter 7, I describe how I adapted the PoseBench benchmark into a deep learning-based ensembling prediction method called MULTICOM_ligand and entered this method as a standalone ligand predictor in the CASP16 competition, ultimately ranking among the top-5 predictors for this category.

Lastly, in Chapter 8, I reflect on the contributions of this dissertation and outline potential future directions for the field of biomolecular modeling and design with deep learning.

# Chapter 2

# GEOMETRIC TRANSFORMERS FOR PROTEIN INTERFACE CONTACT PREDICTION

Adapted from Alex Morehead, Chen Chen, and Jianlin Cheng. "Geometric Transformers for Protein Interface Contact Prediction". *The Tenth International Conference on Learning Representations* (ICLR 2022).

## 2.1 ABSTRACT

Computational methods for predicting the interface contacts between proteins come highly sought after for drug discovery as they can significantly advance the accuracy of alternative approaches, such as protein-protein docking, protein function analysis tools, and other computational methods for protein bioinformatics. In this chapter, we present the Geometric Transformer, a novel geometry-evolving graph transformer for rotation and translation-invariant protein interface contact prediction, packaged within DeepInteract, an end-to-end prediction pipeline. DeepInteract predicts partner-specific protein interface contacts (i.e., inter-protein residue-residue contacts) given the 3D tertiary structures of two proteins as input. In rigorous benchmarks, DeepInteract, on challenging protein complex targets from the 13th and 14th CASP-CAPRI experiments as well as Docking Benchmark 5, achieves 14% and 1.1% top L/5 precision (L: length of a protein unit in a complex), respectively. In doing so, DeepInteract, with the Geometric Transformer as its

graph-based backbone, outperforms existing methods for interface contact prediction in addition to other graph-based neural network backbones compatible with DeepInteract, thereby validating the effectiveness of the Geometric Transformer for learning rich relational-geometric features for downstream tasks on 3D protein structures. Training and inference code as well as pre-trained models are available at `https://github.com/BioinfoMachineLearning/DeepInteract`.

## 2.2 INTRODUCTION

Interactions of proteins, as illustrated in Figure 2.1, often reflect and directly influence their functions in molecular processes, so understanding the relationship between protein interaction and protein function is of utmost importance to biologists and other life scientists. Here, we study the residue-residue interaction between two protein structures that bind together to form a binary protein complex (i.e., dimer), to better understand how these coupled proteins will function *in vivo*. Predicting where two proteins will interface *in silico* has become an appealing method for measuring the interactions between proteins since a computational approach saves time, energy, and resources compared to traditional methods for experimentally measuring such interfaces [33]. A key motivation for determining these interface contacts is to decrease the time required to discover new drugs and to advance the study of newly designed proteins [34].

Existing approaches to interface contact prediction include classical machine learning and deep learning-based methods. These methods traditionally use hand-crafted features to predict which inter-chain pairs of amino acid residues will interact with one another upon the binding of the two protein chains, treating each of their residue pairs as being independent of one another. Recent work on interface prediction [35], however, considers the biological insight that the interaction between two inter-chain residue pairs depends not only on the pairs' features themselves but also on other

residue pairs ordinally nearby in terms of the protein complex's sequence. As such, the problem of interface contact prediction became framed as one akin to image segmentation or object detection, opening the door to innovations in interface contact prediction by incorporating the latest techniques from computer vision.

Nonetheless, up to now, no works on *partner-specific* protein interface contact prediction have leveraged two recent innovations to better capture geometric shapes of protein structures and long-range interactions between amino acids important for accurate prediction of protein-protein interface contacts: (1) geometric deep learning for *evolving* proteins' geometric representations and (2) graph-based self-attention similar to that of [11]. Towards this end, we introduce DeepInteract, an end-to-end deep learning pipeline for protein interface prediction. DeepInteract houses the Geometric Transformer, a new graph transformer designed to exploit protein structure-specific geometric properties, as well as a dilated convolution-based interaction module adapted from [36] to predict which inter-chain residue pairs comprise the interface between the two protein chains. In response to the exponential rate of progress being made in predicting protein structures *in silico*, we trained DeepInteract end-to-end using DIPS-Plus [37], to date the largest feature-rich dataset of protein complex structures for machine learning of protein interfaces, to close the gap on a proper solution to this fundamental problem in structural biology.

### 2.2.1 Related work

Over the past several years, geometric deep learning has become an effective means of automatically learning useful feature representations from structured data [26]. Previously, geometric learning algorithms like convolutional neural networks (CNNs) and graph neural networks (GNNs) have been used to model molecules and to predict protein interface contacts. [39] introduced a deep tensor neural network designed for molecular tasks in quantum chemistry. [40] designed a siamese GNN architecture to

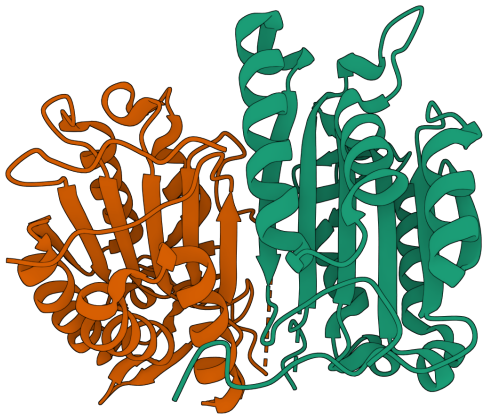


Figure 2.1: A Mol* ([38]) visualization of interacting protein chains (PDB ID: 3H11).

learn weight-tied feature representations of residue pairs. This approach, in essence, processes subgraphs for each residue in each complex and aggregates node-level features locally using a nearest-neighbors approach. Since this *partner-specific* method derives its training dataset from Docking Benchmark 5 (DB5) ([41]), it is ultimately data-limited. [42] represent interacting protein complexes by voxelizing each residue into a 3D grid and encoding in each grid entry the presence and type of the residue's underlying atoms. This *partner-specific* encoding scheme captures static geometric features of interacting complexes, but it is not able to scale well due to its requiring a computationally-expensive spatial resolution of the residue voxels to achieve good results.

Continuing the trend of applying geometric learning to protein structures, [43] developed MaSIF to perform *partner-independent* interface region prediction. Likewise, [44] do so with an attention-based GNN. These methods learn to perform binary classification of the residues in both complex structures to identify regions where residues from both complexes are likely to interact with one another. However, because these approaches predict *partner-independent* interface regions, they are less likely to be useful in helping solve related tasks such as drug-protein interaction prediction and

protein-protein docking [45]. [46] created a graph neural network for predicting the effects of mutations on protein-protein binding affinities, and, more recently, [47] introduced a Euclidean equivariant transformer for protein docking. Both of these methods may benefit from the availability of precise interface predictors by using them to generate contact maps as input features.

To date, one of the best result sets obtained by any model for protein interface contact prediction comes from [35] where high-order (i.e. sequential and coevolution-based) interactions between residues are learned and preserved throughout the network in addition to static geometric features initially embedded in the protein complexes. However, this work, like many of those preceding it, undesirably maintains the trend of reporting model performance in terms of the median area under the receiver operating characteristic which is not robust to extreme class imbalances as often occur in interface contact prediction. In addition, this approach is data-limited as it uses the DB5 dataset and its predecessors to derive both its training data and makes use of only each residue's carbon-alpha (C$\alpha$) atom in deriving its geometric features, ignoring important geometric details provided by an all-atom view of protein structures.

Our work builds on top of prior works by making the following contributions:

- We provide the *first* example of graph self-attention applied to protein interface contact prediction, showcasing its effective use in learning representations of protein geometries to be exploited in downstream tasks.

- We propose the new Geometric Transformer which can be used for tasks on 3D protein structures and similar biomolecules. For the problem of interface contact prediction, we train the Geometric Transformer to evolve a geometric representation of protein structures simultaneously with protein sequence and coevolutionary features for the prediction of inter-chain residue-residue contacts. In doing so, we also demonstrate the merit of the Enhanced Database of

Interacting Protein Structures (DIPS-Plus) for interface prediction [37].

- Our experiments on challenging protein complex targets demonstrate that our proposed method, DeepInteract, achieves state-of-the-art results for interface contact prediction.

## 2.3 METHODS

### 2.3.1 Datasets

The current opinion in the bioinformatics community is that protein sequence features still carry important higher-order information concerning residue-residue interactions [35]. In particular, the residue-residue coevolution and residue conservation information obtained through multiple sequence alignments (MSAs) has been shown to contain powerful information concerning intra-chain and even inter-chain residue-residue interactions as they yield a compact representation of residues' coevolutionary relationships [15].

Keeping this in mind, for our training and validation datasets, we chose to use DIPS-Plus [37], one of the largest feature-rich datasets of protein complexes for protein interface contact prediction. In total, DIPS-Plus contains 42,112 binary protein complexes with positive labels (i.e., 1) for each inter-chain residue pair that are found within 6 Å of each other in the complex's bound (i.e., structurally-conformed) state. The dataset contains a variety of rich residue-level features: (1) an 8-state one-hot encoding of the secondary structure in which the residue is found; (2) a scalar solvent accessibility; (3) a scalar residue depth; (4) a $1 \times 6$ vector detailing each residue's protrusion concerning its side chain; (5) a $1 \times 42$ vector describing the composition of amino acids towards and away from each residue's side chain; (6) each residue's coordinate number conveying how many residues to which the residue meets a significance threshold, (7) a $1 \times 27$ vector giving residues' emission and transition probabilities

derived from HH-suite3 [48] profile hidden Markov models constructed using MSAs; and (8) amide plane normal vectors for downstream calculation of the angle between each intra-chain residue pair's amide planes.

To compare the performance of DeepInteract with that of state-of-the-art methods, we select 32 homodimers and heterodimers from the test partition of DIPS-Plus to assess each method's competency in predicting interface contacts. We also evaluate each method on 14 homodimers and 5 heterodimers with PDB structures publicly available from the 13th and 14th sessions of CASP-CAPRI [49, 50] as these targets are considered by the bioinformatics community to be challenging for existing interface predictors. For any CASP-CAPRI test complexes derived from multimers (i.e., protein complexes that can contain more than two chains), to represent the complex we chose the pair of chains with the largest number of interface contacts. Finally, we use the traditional 55 test complexes from the DB5 dataset [40, 42, 35] to benchmark each heteromer-compatible method.

To expedite training and validation and to constrain memory usage, beginning with all remaining complexes not chosen for testing, we filtered out all complexes where either chain contains fewer than 20 residues and where the number of possible interface contacts is more than $256^2$, leaving us with an intermediate total of 26,504 complexes for training and validation. In the initial version of DIPS-Plus which we adopted in this work, binary protein complexes are grouped into shared directories according to whether they are derived from the same parent complex. As such, using a *per-directory* strategy, we randomly designate 80% of these complexes for training and 20% for validation to restrict overlap between our cross-validation datasets. After choosing these targets for testing, we then filter out complexes from our training and validation partitions of DIPS-Plus that contain any chain with over 30% sequence identity to any chain in any complex in our test datasets. This threshold of 30% sequence identity is commonly used in the bioinformatics literature [51, 52] to prevent

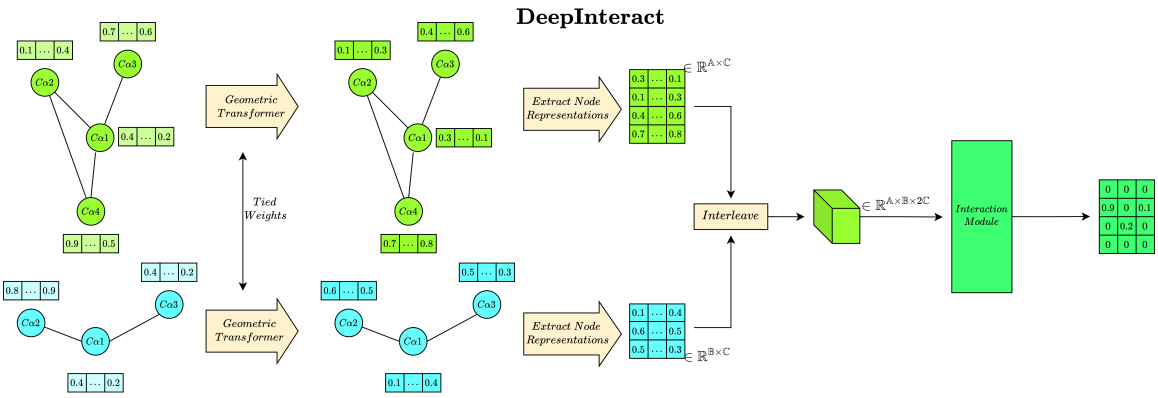



Figure 2.2: A framework overview of DeepInteract. The proposed pipeline separates interface contact prediction into two tasks: (1) learning new node representations $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$ for pairs of residue protein graphs and (2) convolving over $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$ interleaved together to predict pairwise contact probabilities.

large evolutionary overlap between a dataset's cross-validation partitions. However, most existing works for interface contact prediction do not employ such filtering criteria, so the results reported in these works may be over-optimistic by nature. In performing such sequence-based filtering, we are left with 15,618 and 3,548 binary complexes for training and validation, respectively.

### 2.3.2 Problem formulation

Summarized in Figure 2.2, we designed DeepInteract, our proposed pipeline for interface contact prediction, to frame the problem of predicting interface contacts *in silico* as a two-part task: The first part is to use attentive graph representation learning to inductively learn new node-level representations $h_{\mathbb{A}} \in \mathbb{R}^{\mathbb{A}\times\mathbb{C}}$ and $h_{\mathbb{B}} \in \mathbb{R}^{\mathbb{B}\times\mathbb{C}}$ for a pair of graphs representing two protein chains. The second part is to channel-wise interleave $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$ into an interaction tensor $\mathbb{I} \in \mathbb{R}^{\mathbb{A}\times\mathbb{B}\times 2\mathbb{C}}$, where $\mathbb{A} \in \mathbb{R}$ and $\mathbb{B} \in \mathbb{R}$ are the numbers of amino acid residues in the first and second input protein chains, respectively, and $\mathbb{C} \in \mathbb{R}$ is the number of hidden channels in both $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$. We use interaction tensors such as $\mathbb{I}$ as input to our interaction module, a convolution-based dense predictor of inter-graph node-node interactions.

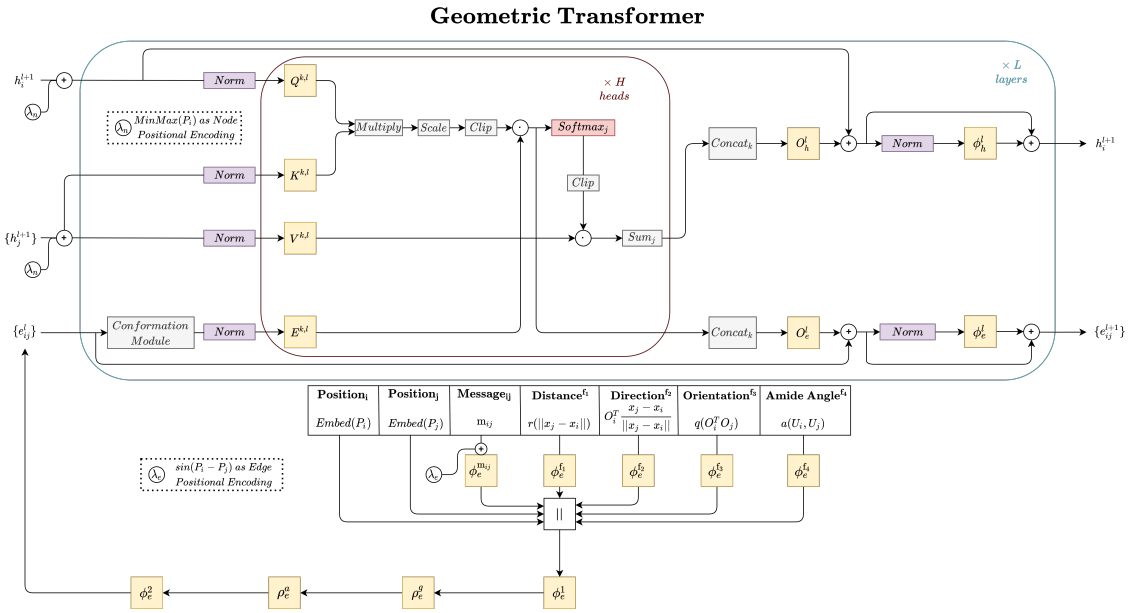



Figure 2.3: An overview of the GEOMETRIC TRANSFORMER. Notably, the final layer of the GEOMETRIC TRANSFORMER removes the edge update path since, in this formulation of interface prediction, only graph pairs' node representations $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$ are directly used for the final interface contact prediction.

We denote each protein chain in an input complex as a graph $\mathbb{G}$ with edges $\mathbb{E}$ between the $k$-nearest neighbors of its nodes $\mathbb{N}$, with nodes corresponding to the chain's amino acid residues represented by their C$\alpha$ atoms. In this setting, we let $k = 20$ as we observed favorable cross entropy loss on our validation dataset with this level of connectivity. We note that this level of graph connectivity has also proven to be advantageous for prior works developing deep learning approaches for graph-based protein representations [40, 53].

### 2.3.3 Geometric Transformer architecture

Hypothesizing that a self-attention mechanism that evolves proteins' physical geometries is a key component missing from existing interface contact predictors, we propose the GEOMETRIC TRANSFORMER, a graph neural network explicitly designed for capturing and iteratively *evolving* protein geometric features. As shown in Figure 2.3, the GEOMETRIC TRANSFORMER builds upon the existing Graph Transformer

architecture [54] by introducing (**1**) an edge initialization module, (**2**) an edge-wise positional encoding (EPE), and (**3**) a geometry-evolving conformation module employing repeated geometric feature gating (GFG) (see Appendices A.6, A.7, and A.8 for rationale). Moreover, the GEOMETRIC TRANSFORMER includes subtle architectural enhancements to the original Transformer architecture [11] such as moving the network's first normalization layer to precede any affinity score computations for improved training stability [55]. To our knowledge, the GEOMETRIC TRANSFORMER is the *first* deep learning model that applies multi-head attention to the task of *partner-specific* protein interface prediction. The following sections serve to distinguish our new GEOMETRIC TRANSFORMER from other Transformer-like architectures by describing its new neural network modules for geometric self-attention.

### 2.3.4 Edge initialization module

To enrich its expressivity, the GEOMETRIC TRANSFORMER first embeds each edge $e \in \mathbb{E}$ with the initial edge representation

$$c_{ij} = \phi_e^1([\mathrm{p}_1 \,||\, \mathrm{p}_2 \,||\, \phi_e^{\mathrm{m}_{ij}}(\mathrm{m}_{ij} \,||\, \lambda_e) \,||\, \phi_e^{\mathrm{f}_1}(\mathrm{f}_1) \,||\, \phi_e^{\mathrm{f}_2}(\mathrm{f}_2) \,||\, \phi_e^{\mathrm{f}_3}(\mathrm{f}_3) \,||\, \phi_e^{\mathrm{f}_4}(\mathrm{f}_4)]) \tag{2.1}$$

$$e_{ij} = \phi_e^2(\rho_e^a(\rho_e^g(c_{ij}))) \tag{2.2}$$

where $\phi_e^i$ refers to the $i$'th edge information update function such as a multi-layer perceptron; $||$ denotes channel-wise concatenation; $\mathrm{p}_1$ and $\mathrm{p}_2$, respectively, are trainable one-hot vectors indexed by $P_i$ and $P_j$, the positions of nodes $i$ and nodes $j$ in the chain's underlying amino acid sequence; $\mathrm{m}_{ij}$ are any user-predefined features for $e$ (in our case the normalized Euclidean distances between nodes $i$ and nodes $j$); $\lambda_e$ are *edge-wise* sinusoidal positional encodings $sin(P_i - P_j)$ for $e$; $\mathrm{f}_1$, $\mathrm{f}_2$, $\mathrm{f}_3$, and $\mathrm{f}_4$, in order, are the four protein-specific geometric features defined in Appendix A.3; and

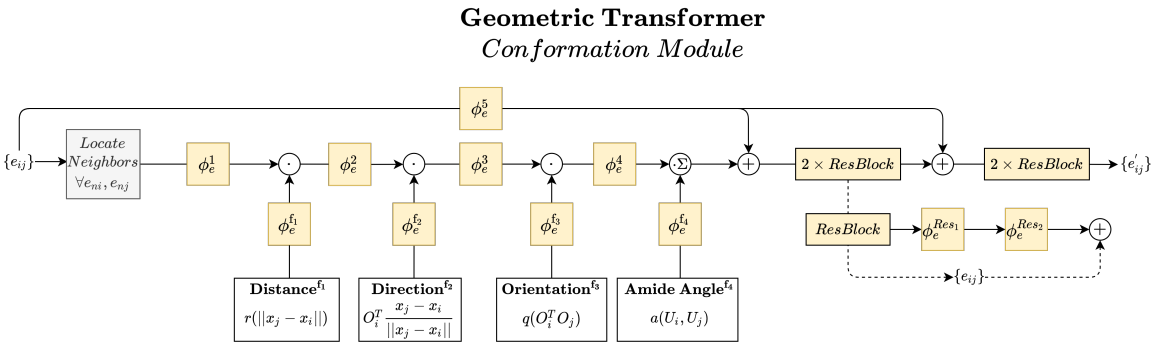



Figure 2.4: An overview of the Conformation Module. The GEOMETRIC TRANSFORMER uses a Conformation Module in each layer to evolve protein graphs' geometric representations via repeated gating and a final series of residual connection blocks.

$\rho_e^a$ and $\rho_e^g$ are feature addition and channel-wise gating functions, respectively.

### Conformation Module

The role of the GEOMETRIC TRANSFORMER's subsequent conformation module, as illustrated in Figure 2.4, is for it to learn how to iteratively *evolve* geometric representations of protein graphs by applying repeated gating to our initial edge geometric features $f_1$, $f_2$, $f_3$, and $f_4$. To do so, the conformation module updates $e_{ij}$ by introducing the notion of a *geometric neighborhood* of edge $e$, treating $e$ as a pseudo-node. Precisely, $\mathbb{E}_k$, the edge geometric neighborhood of $e$, is defined as the $2n$ edges

$$\mathbb{E}_k = \{e_{n_1 i}, e_{n_2 j} \mid (n_1, n_2 \in \mathbb{N}_k) \text{ and } (n_1, n_2 \neq i, j)\}, \tag{2.3}$$

where $\mathbb{N}_k \subset \mathbb{N}$ are the source nodes for incoming edges on edge $e's$ source and destination nodes. The intuition behind updating each edge according to its $2n$ nearest neighboring edges is that the geometric relationship between a residue pair, described by their mutual edge's features, can be influenced by the physical constraints imposed by proximal residue-residue geometries. As such, we use these nearby edges during geometric feature updates. In the conformation module, the iterative processing of all geometric neighborhood features for edge $e$ can be represented as

$$O_{ij} = \sum_{k \in \mathbb{E}_k} [(\phi_e^n(e_{ij,k}^n) \odot \phi_e^{\mathrm{f}_n}(\mathrm{f}_n)), \forall n \in \mathbb{F}] \tag{2.4}$$

$$e_{ij} = 2 \times ResBlock_2(\phi_e^5(e_{ij}) + 2 \times ResBlock_1(\phi_e^5(e_{ij}) + O_{ij})), \tag{2.5}$$

where $\mathbb{F}$ are the indices of the geometric features $\{\mathrm{f}_1, \mathrm{f}_2, \mathrm{f}_3, \mathrm{f}_4\}$ defined in Appendix A.3; $\odot$ is element-wise multiplication; $e_{ij,k}^n$ is neighboring edge $e_k$'s representation after gating with $\mathrm{f}_{\mathrm{n}-1}$; and $2 \times ResBlock_i$ represents the $i$'th application of two unique, successive residual blocks, each defined as $ResBlock(x) = \phi_e^{Res_2}(\phi_e^{Res_1}(x)) + x$. Described in Section A.3, by way of their construction, each of our selected edge geometric features is translation and rotation invariant to the network's input space. As discussed in Appendix A.5, we couple these features with our choice of node-wise positional encodings (see Section 2.3.5) to attain canonical invariant local frames for each residue to encode the relative poses of features in our protein graphs. In doing so, we leverage many of the benefits of employing equivariant representations while reducing the large memory requirements they typically induce, to yield a robust invariant representation of each input protein.

### 2.3.5 Selected Transformer initializations and operations

For the initial node features used within the Geometric Transformer, we include each of DIPS-Plus' residue-level features described succinctly in Section 2.3.1. Additionally, we append initial min-max normalizations of each residue's index in $P_i$ to each node as node-wise positional encodings. For the remainder of the Geometric Transformer's operations, the network's order of operations closely follows the definitions given by [54] for the Graph Transformer, with an exception being that the first normalization layer now precedes any affinity score calculations.

### 2.3.6 Interaction module

Upon applying multiple layers of the Geometric Transformer to each pair of input protein chains, we then channel-wise interleave the Geometric Transformer's learned node representations $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$ into $\mathbb{I}$ to serve as input to our interaction module, consisting of a dilated ResNet module adapted from [36]. The core residual network component in this interaction module consists of four residual blocks differing in the number of internal layers. Each residual block is comprised of several consecutive instance normalization layers and convolutional layers with 64 kernels of size $3 \times 3$. The number of layers in each block represents the number of 2D convolution layers in the corresponding component. The final values of the last convolutional layer are added to the output of a shortcut block, which is a convolutional layer with 64 kernels of size $1 \times 1$. A squeeze-and-excitation (SE) block [56] is added at the end of each residual block to adaptively recalibrate its channel-wise feature responses. Ultimately, the output of the interaction module is a probability-valued $\mathbb{A}$ x $\mathbb{B}$ matrix that can be viewed as an inter-chain residue binding heatmap.

## 2.4 RESULTS

### 2.4.1 Experiments

#### Setup

For all experiments conducted with DeepInteract, we used 2 layers of the graph neural network chosen for the experiment and 128 intermediate GNN and CNN channels to restrict the time required to train each model. For the Geometric Transformer, we used an edge geometric neighborhood of size $n = 2$ for each edge such that each edge's geometric features are updated by their 4-nearest incoming edges. In addition, we used the Adam optimizer [57], a learning rate of $1e^{-3}$, a weight decay rate

of $1e^{-2}$, a dropout (i.e., forget) rate of 0.2, and a batch size of 1. We also employed 0.5-threshold gradient value clipping and stochastic weight averaging [58]. With an early-stopping patience period of 5 epochs, we observed most models converging after approximately 30 training epochs on DIPS-Plus. For our loss function, we used weighted cross entropy with a positive class weight of 5 to help the network overcome the large class imbalance present in interface prediction. All DeepInteract models employed 14 layers of our dilated ResNet architecture described in Section 2.3.6 and had their top-$k$ metrics averaged over three separate runs, each with a different random seed (standard deviation of top-$k$ metrics in parentheses). Prior to our experiments on the DB5 dataset's 55 test complexes, we fine-tuned each DeepInteract model using the held-out 140 and 35 complexes remaining in DB5 for training and validation, respectively. Employing a similar training configuration as described above, in this context we used a lower learning rate of $1e^{-5}$ to facilitate smoother transfer learning between DIPS-Plus and DB5.

### Hyperparameter search

To identify our optimal set of model hyperparameters, we performed a manual hyperparameter search over a learning rate range of $[1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}, 1e^{-5}, 1e^{-6}]$ and a weight decay rate range of $[1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}]$, respectively. In doing so, we found a learning rate of $1e^{-3}$ and a weight decay rate of $1e^{-2}$ to provide the lowest loss and the highest metric values on our DIPS-Plus validation dataset. We restricted our hyperparameter search to the learning rate and weight decay rate of our models due to the large computational and environmental costs associated with training each model. However, this suggests further improvements to our models could be found with a more extensive hyperparameter search over, for example, the models' dropout rate.

Table 2.1: The average top-$k$ precision on two types of DIPS-Plus test targets.

| | 16 (Homo) | | | 16 (Hetero) | | |
|---|---|---|---|---|---|---|
| Method | 10 | $L/10$ | $L/5$ | 10 | $L/10$ | $L/5$ |
| BI | 0 | 0 | 0 | 0.02 | 0.02 | 0.02 |
| DH | 0.13 | 0.12 | 0.09 | | | |
| CC | | | | **0.17** | **0.16** | **0.15** |
| DI (GCN) | 0.22 (0.06) | 0.20 (0.07) | 0.18 (0.04) | 0.08 (0.01) | 0.08 (0.01) | 0.07 (0.02) |
| DI (GT) | 0.27 (0.06) | 0.24 (0.04) | 0.21 (0.04) | 0.10 (0.04) | 0.09 (0.04) | 0.08 (0.04) |
| DI (GeoT w/o EPE) | **0.28** (**0.05**) | 0.24 (0.01) | 0.23 (0.03) | 0.11 (0.05) | 0.10 (0.04) | 0.09 (0.03) |
| DI (GeoT w/o GFG) | 0.27 (0.08) | 0.24 (0.08) | 0.21 (0.08) | 0.10 (0.02) | 0.09 (0.02) | 0.09 (0.01) |
| DI (GeoT) | 0.25 (0.03) | **0.25** (**0.03**) | **0.23** (**0.02**) | 0.15 (0.04) | 0.14 (0.05) | 0.11 (0.04) |

Table 2.2: The average top-$k$ precision and recall on DIPS-Plus test targets of both types.

| | 32 (Both Types) | | | | | |
|---|---|---|---|---|---|---|
| Method | P@10 | P@$L/10$ | P@$L/5$ | R@$L$ | R@$L/2$ | R@$L/5$ |
| BI | 0.01 | 0.01 | 0.01 | 0.01 | 0.004 | 0.003 |
| DI (GCN) | 0.15 (0.03) | 0.16 (0.01) | 0.12 (0.02) | 0.10 (0.02) | 0.06 (0.01) | 0.03 (0.003) |
| DI (GT) | 0.18 (0.05) | 0.16 (0.04) | 0.15 (0.04) | 0.13 (0.02) | 0.07 (0.01) | 0.04 (0.01) |
| DI (GeoT w/o EPE) | 0.19 (0.04) | 0.18 (0.03) | 0.16 (0.03) | 0.14 (0.02) | 0.08 (0.02) | 0.04 (0.02) |
| DI (GeoT w/o GFG) | 0.18 (0.05) | 0.16 (0.04) | 0.15 (0.04) | 0.14 (0.02) | 0.08 (0.02) | 0.04 (0.01) |
| DI (GeoT) | **0.20** (**0.01**) | **0.19** (**0.01**) | **0.17** (**0.02**) | **0.15** (**0.003**) | **0.09** (**0.004**) | **0.04** (**0.002**) |

### Selection of baselines

We considered the reproducibility and accessibility of a method to be the most important factors for its inclusion in our following benchmarks to encourage the adoption of accessible and transparent benchmarks for future works. As such, we have included the methods BIPSPI (an XGBoost-based algorithm) [59], DeepHomo (a CNN for homodimers) [60], and ComplexContact (a CNN for heterodimers) [61] since they are either easy to reproduce or simple for the general public to use to make predictions. Each method predicts interfacing residue pairs subject to the (on average) 1:1000 positive-negative class imbalance imposed by the biological sparsity of true interface contacts. We note that we also considered adding more recent baseline methods such as those of [42] and [35]. However, for both of these methods, we were not able to locate any provided source code or web server predictors facilitating the prediction

Table 2.3: The average top-$k$ precision on dimers from CASP-CAPRI 13 & 14.

| | 14 (Homo) | | | 5 (Hetero) | | |
|---|---|---|---|---|---|---|
| Method | 10 | $L/10$ | $L/5$ | 10 | $L/10$ | $L/5$ |
| BI | 0 | 0 | 0 | 0.04 | 0 | 0.03 |
| DH | 0.02 | 0.02 | 0.02 | | | |
| CC | | | | 0.06 | 0.08 | 0.05 |
| DI (GCN) | 0.12 (0.04) | 0.11 (0.03) | **0.13** (**0.02**) | 0.10 (0.07) | 0.11 (0.08) | 0.09 (0.04) |
| DI (GT) | 0.08 (0.03) | 0.09 (0.05) | 0.08 (0.03) | 0.14 (0.02) | 0.14 (0.02) | 0.12 (0.03) |
| DI (GeoT w/o EPE) | 0.11 (0.01) | 0.12 (0.02) | 0.11 (0.01) | 0.18 (0.07) | 0.20 (0.09) | 0.18 (0.04) |
| DI (GeoT w/o GFG) | 0.10 (0.02) | 0.10 (0.02) | 0.09 (0.02) | 0.14 (0.03) | 0.17 (0.03) | 0.14 (0.02) |
| DI (GeoT) | **0.18** (**0.05**) | **0.13** (**0.03**) | 0.11 (0.02) | **0.30** (**0.09**) | **0.31** (**0.07**) | **0.24** (**0.04**) |

Table 2.4: The average top-$k$ precision and recall across all targets from CASP-CAPRI 13 & 14.

| | 19 (Both Types) | | | | | |
|---|---|---|---|---|---|---|
| Method | P@10 | P@$L/10$ | P@$L/5$ | R@$L$ | R@$L/2$ | R@$L/5$ |
| BI | 0.01 | 0 | 0.01 | 0.02 | 0.01 | 0.001 |
| DI (GCN) | 0.12 (0.04) | 0.10 (0.05) | 0.09 (0.04) | 0.11 (0.001) | 0.06 (0.01) | 0.02 (0.01) |
| DI (GT) | 0.10 (0.03) | 0.09 (0.03) | 0.08 (0.02) | 0.11 (0.02) | 0.06 (0.01) | 0.02 (0.01) |
| DI (GeoT w/o EPE) | 0.13 (0.02) | 0.14 (0.03) | 0.13 (0.02) | 0.12 (0.01) | 0.07 (0.01) | 0.03 (0.01) |
| DI (GeoT w/o GFG) | 0.11 (0.01) | 0.12 (0.02) | 0.10 (0.02) | 0.11 (0.01) | 0.06 (0.01) | 0.03 (0.01) |
| DI (GeoT) | **0.21** (**0.01**) | **0.19** (**0.01**) | **0.14** (**0.01**) | **0.13** (**0.02**) | **0.08** (**0.01**) | **0.04** (**0.003**) |

of inter-protein residue-residue contacts for provided FASTA or PDB targets, so they ultimately did not meet our baseline selection criterion of reproducibility (i.e., an ability to make new predictions). We also include two ablation studies (e.g., DI (GeoT w/o GFG)) to showcase the effect of including network components unique to the Geometric Transformer.

Table 2.5: The average top-$k$ precision and recall on DB5 test targets.

| | 55 (Hetero) | | | | | |
|---|---|---|---|---|---|---|
| Method | P@10 | P@$L/10$ | P@$L/5$ | R@$L$ | R@$L/2$ | R@$L/5$ |
| BI | 0 | 0.002 | 0.001 | 0.003 | 0.001 | 0.0004 |
| CC | 0.002 | 0.003 | 0.003 | 0.007 | 0.003 | 0.001 |
| DI (GCN) | 0.005 (0.002) | 0.006 (0.001) | 0.007 (0.001) | 0.013 (0.002) | 0.008 (0.001) | 0.003 (0.001) |
| DI (GT) | 0.008 (0.004) | 0.008 (0.005) | 0.008 (0.004) | 0.010 (0.005) | 0.006 (0.003) | 0.003 (0.002) |
| DI (GeoT w/o EPE) | 0.011 (0.004) | 0.009 (0.004) | 0.011 (0.002) | 0.018 (0.01) | 0.010 (0.004) | 0.0034 (0.002) |
| DI (GeoT w/o GFG) | 0.008 (0.001) | 0.008 (0.001) | 0.009 (0.002) | 0.014 (0.01) | 0.006 (0.002) | 0.003 (0.001) |
| DI (GeoT) | **0.013** (**0.001**) | **0.009** (**0.003**) | **0.011** (**0.001**) | **0.018** (**0.001**) | **0.010** (**0.001**) | **0.0034** (**0.001**) |

Our selection criterion for each baseline method consequently determined the number of complexes against which we could feasibly test each method, thereby restricting the size of our test datasets to 106 complexes in total. In addition, not all baselines chosen were originally trained for both types of protein complexes (i.e., homodimers and heterodimers), so for these baselines we do not include their results for the type of complex for which they are not respectively designed.

For brevity, in all experiments, we refer to BIPSPI, DeepHomo, ComplexContact, and DeepInteract as BI, DH, CC, and DI, respectively. Further, we refer to the Graph Convolutional Network of [62], the Graph Transformer of [54], and the Geometric Transformer as GCN, GT, and GeoT, respectively. To assess the models' ability to correctly select residue pairs in interaction upon binding of two given chains, all methods are scored using the top-$k$ precision and recall metrics (defined in Appendix A.2) commonly used for intra-chain contact prediction [36] as well as recommender systems [63], where $k \in \{10, L/10, L/5, L/2\}$ with $L$ being the length of the shortest chain in a given complex.

## 2.5 DISCUSSION

Table 2.1 demonstrates that DeepInteract outperforms or achieves competitive results compared to existing state-of-the-art methods for interface contact prediction on DIPS-Plus with both types of protein complexes, homodimers (homo) where the two chains are of the same protein and heterodimers (hetero) where the two chains are of different proteins. Table 2.2 shows that, when taking both types of complexes into account, DeepInteract outperforms all other methods' predictions on DIPS-Plus. Since future users of DeepInteract may want to predict interface contacts for either type of complex, we consider a method's type-averaged top-$k$ metrics as important metrics for which to optimize.

Likewise, Tables 2.3 and 2.4 present the average top-$k$ metrics of DeepInteract

on 19 challenging protein complexes (14 homodimers and 5 heterodimers) from the 13th and 14th rounds of the joint CASP-CAPRI meeting. In them, we once again see DeepInteract exceed the precision of state-of-the-art interface contact predictors for both complex types. In particular, we see that combining DeepInteract with the Geometric Transformer offers improvements to the majority of our top-$k$ metrics for both homodimers and heterodimers compared to using either a GCN or a Graph Transformer-based GNN backbone, notably for heteromeric complexes with largely asymmetric inter-chain geometries. Such a result supports our hypothesis that the Geometric Transformer's geometric self-attention mechanism can enable enhanced prediction performance for downstream tasks on geometrically-intricate 3D objects such as protein structures, using interface contact prediction as a case study.

Finally, in Table 2.5, we observe that, in predicting the interface contacts between *unbound* protein chains in the DB5 test dataset, the Geometric Transformer enables enhanced top-$k$ precision and recall (definition in A.2) compared to all other baseline methods, including GCNs and Graph Transformers paired with DeepInteract. Such a result confirms, to a moderate degree, the Geometric Transformer's ability to predict how the structural conformations occurring upon the binding of two protein chains influence which inter-chain residue pairs will interact with one another in the complex's *bound* state.

### 2.5.1 Conclusions

In this chapter, we introduced DeepInteract which debuts the geometry-evolving Geometric Transformer for protein representation learning and demonstrates its effectiveness in predicting protein-protein interactions. We envision several other uses of the Geometric Transformer in protein deep learning such as quaternary structure quality assessment [64] and residue disorder prediction. One limitation of the Geometric Transformer's design is its lack of equivariant representations for

coordinates-based prediction tasks, which we aim to address in the next chapter.

# Chapter 3

# GEOMETRY-COMPLETE PERCEPTRON NETWORKS FOR 3D MOLECULAR GRAPHS

Adapted from Alex Morehead and Jianlin Cheng. "Geometry-complete perceptron networks for 3D molecular graphs". *Bioinformatics* 40.2 (2024): btae087.

## 3.1 ABSTRACT

The field of geometric deep learning has recently had a profound impact on several scientific domains such as protein structure prediction and design, leading to methodological advancements within and outside of the realm of traditional machine learning. Within this spirit, in this chapter, we introduce GCPNet, a new chirality-aware SE(3)-equivariant graph neural network designed for representation learning of 3D biomolecular graphs. We show that GCPNet, unlike previous representation learning methods for 3D biomolecules, is widely applicable to a variety of invariant or equivariant node-level, edge-level, and graph-level tasks on biomolecular structures while being able to (1) learn important chiral properties of 3D molecules and (2) detect external force fields. Across four distinct molecular-geometric tasks, we demonstrate that GCPNet's predictions (1) for protein-ligand binding affinity achieve a statistically significant correlation of 0.608, more than 5% greater than current state-of-the-art methods; (2) for protein structure ranking achieve statistically significant target-local and dataset-global correlations of 0.616 and 0.871, re-

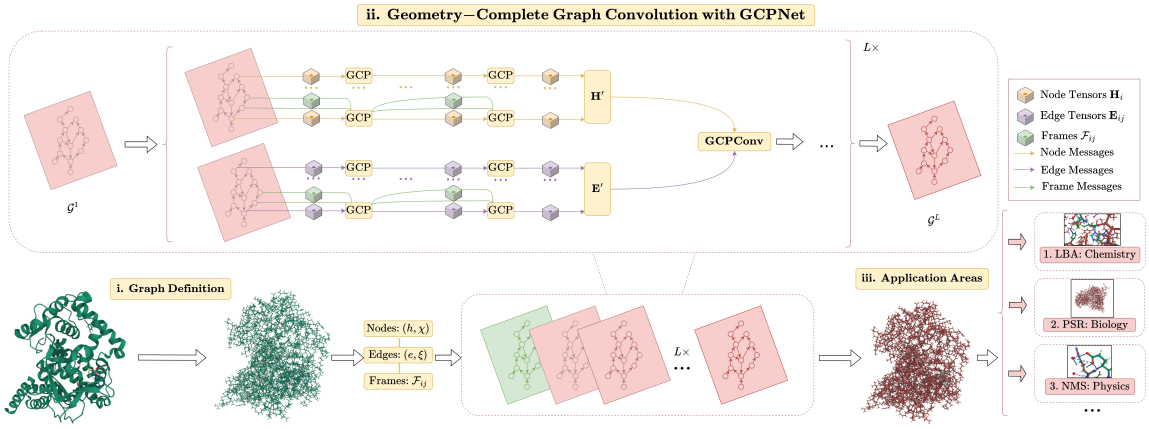



Figure 3.1: A framework overview for our proposed *Geometry-Complete Perceptron Network* (GCPNet). Our framework consists of (**i.**) a graph (topology) definition process, (**ii.**) a GCPNet-based graph neural network for 3D molecular representation learning, and (**iii.**) demonstrated application areas for GCPNet.

spectively; (3) for Newtownian many-body systems modeling achieve a task-averaged mean squared error less than 0.01, more than 15% better than current methods; and (4) for molecular chirality recognition achieve a state-of-the-art prediction accuracy of 98.7%, better than any other machine learning method to date. The source code, data, and instructions to train new models or reproduce our results are freely available at `https://github.com/BioinfoMachineLearning/GCPNet`.

## 3.2 INTRODUCTION

Over the last several years, the field of deep learning has pioneered many new methods designed to process graph-structured inputs. Being a ubiquitous form of information, graph-structured data arises from numerous sources such as the fields of physics and chemistry, for example in the form of interacting particle systems or molecular graphs. Moreover, the relational nature of graph-structured data allows one to identify and characterize topological associations between entities in large real-world networks (e.g., social networks).

In scientific domains such as computational biology and chemistry, graphs are

often used to represent the 3D structures of molecules [65], chemical compounds [66], and even large biomolecules such as proteins [67, 68, 69, 70, 71]. Underlying many of these successful examples of graph representations are graph neural networks (GNNs), a class of machine learning algorithms specialized in processing irregularly-structured input data such as graphs. Careful applications of graph neural networks in scientific domains have considered the physical symmetries present in many scientific data and have leveraged such symmetries to design new attention-based neural network architectures [72, 15].

Throughout their development, geometric deep learning methods have expanded to incorporate within them equivariance to various geometric symmetry groups to enhance their generalization capabilities and adversarial robustness. Methods such as group-equivariant CNNs [73], Tensor Field Networks [74], and equivariant GNNs [75] such as GVP-GNNs [17, 76] and ClofNet [77] have paved the way for the development of future deep learning models that respect physical symmetries present in 3D data (e.g., rotation equivariance with respect to input data symmetries).

Within this spirit, in this work, we introduce a new geometric graph neural network model, GCPNet, that is equivariant to the group of 3D rotations and translations (i.e., SE(3), the special Euclidean group, as studied in previous works [78]) and, uniquely, that simultaneously guarantees chirality sensitivity and geometric (vector) information completeness following graph message-passing on 3D point clouds. We demonstrate its expressiveness and flexibility for modeling physical systems through rigorous experiments for distinct molecular-geometric tasks. In detail, we provide the following contributions:

- In contrast to prior geometric networks for molecules that are insensitive to their chemical chirality [17, 76], cannot detect global physical forces acting upon each atom [79], or do not directly learn geometric features [77], we present the first geometric graph neural network architecture with the following desirable

properties for learning from 3D molecules as described in Appendix B.4.1: (1) the ability to directly predict translation and rotation-invariant scalar properties and rotation-equivariant vector-valued quantities for nodes and edges, respectively; (2) a rotation and translation-equivariant method for iteratively updating node positions in 3D space; (3) sensitivity to molecular chirality; and (4) a means by which to learn from and account for the global forces acting upon the atoms within its inputs.

- We establish new state-of-the-art results for four distinct molecular-geometric representation learning tasks - molecular chirality recognition, protein-ligand binding affinity prediction, protein structure ranking, and Newtonian many-body-systems modeling - where model predictions vary from analyzing individual nodes to summarizing entire graph inputs. GCPNet's performance for these tasks is statistically significant and surpasses that of previous state-of-the-art machine learning methods for 3D molecules.

## 3.3 METHODS

### 3.3.1 Preliminaries

#### Overview of the problem setting

We represent a 3D molecular structure (e.g., a protein or small molecule) as a 3D $k$-nearest neighbors ($k$-NN) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ and $\mathcal{E}$ representing the graph's set of nodes and set of edges, respectively, and $N = |\mathcal{V}|$ and $E = |\mathcal{E}|$ representing the number of nodes and the number of edges in the graph, respectively. In addition, $\mathbf{X} \in \mathbb{R}^{N \times 3}$ represents the respective Cartesian coordinates for each node. We then design E(3)-invariant (i.e., 3D rotation, reflection, and translation-invariant) node features $\mathbf{H} \in \mathbb{R}^{N \times h}$ and edge features $\mathbf{E} \in \mathbb{R}^{E \times e}$ as well as O(3)-equivariant (3D rotation and reflection-equivariant) node features $\boldsymbol{\chi} \in \mathbb{R}^{N \times (m \times 3)}$ and edge features

$\boldsymbol{\xi} \in \mathbb{R}^{E\times(x\times 3)}$, respectively.

Upon constructing such features, we apply several layers of graph message-passing using a neural network $\boldsymbol{\Phi}$ (which later on we refer to as GCPNet) that updates node and edge features using invariant and equivariant representations for the corresponding feature types. Importantly, $\boldsymbol{\Phi}$ guarantees, by design, *SE(3) equivariance* with respect to its vector-valued input coordinates and features (i.e., $x_i \in \mathbf{X}$, $\chi_i \in \boldsymbol{\chi}$, and $\xi_{ij} \in \boldsymbol{\xi}$) and *SE(3)-invariance* regarding its scalar features (i.e., $h_i \in \mathbf{H}$ and $e_{ij} \in \mathbf{E}$). In addition to SE(3) equivariance, $\boldsymbol{\Phi}$'s scalar graph representations achieve *geometric self-consistency* and *geometric completeness* for the 3D structure of the input molecular graph $\mathcal{G}$ as formalized in the definitions below, where $\square'$ represents an updated feature.

**Definition 1.** *(SE(3) Equivariance).*
*Given* $(\mathbf{H}', \mathbf{E}', \mathbf{X}', \boldsymbol{\chi}', \boldsymbol{\xi}') = \boldsymbol{\Phi}(\mathbf{H}, \mathbf{E}, \mathbf{X}, \boldsymbol{\chi}, \boldsymbol{\xi})$*, we have*
$(\mathbf{H}', \mathbf{E}', \mathbf{Q}\mathbf{X}'^{\mathrm{T}} + \mathbf{g}, \mathbf{Q}\boldsymbol{\chi}'^{\mathrm{T}}, \mathbf{Q}\boldsymbol{\xi}'^{\mathrm{T}}) = \boldsymbol{\Phi}(\mathbf{H}, \mathbf{E}, \mathbf{Q}\mathbf{X}^{\mathrm{T}} + \mathbf{g}, \mathbf{Q}\boldsymbol{\chi}^{\mathrm{T}}, \mathbf{Q}\boldsymbol{\xi}^{\mathrm{T}})$*,*
$\forall \mathbf{Q} \in SO(3), \forall \mathbf{g} \in \mathbb{R}^{3\times 1}$.

**Definition 2.** *(Geometric Self-Consistency).*
*Given a pair of molecular graphs* $\mathcal{G}_1$ *and* $\mathcal{G}_2$*,*
*with* $\mathbf{X}^1 = \{\mathbf{x}_i^1\}_{i=1,\ldots,N}$ *and* $\mathbf{X}^2 = \{\mathbf{x}_i^2\}_{i=1,\ldots,N}$*, respectively,*
*a geometric representation* $\boldsymbol{\Phi}(\mathbf{H}, \mathbf{E}) = \boldsymbol{\Phi}(\mathcal{G})$ *is considered*
*geometrically self-consistent if* $\boldsymbol{\Phi}(\mathcal{G}^1) = \boldsymbol{\Phi}(\mathcal{G}^2) \iff \exists \mathbf{Q} \in SO(3), \exists \mathbf{g} \in \mathbb{R}^{3\times 1}$*,*
*for* $i = 1, \ldots, n, \mathbf{X}_i^{1^{\mathrm{T}}} = \mathbf{Q}\mathbf{X}_i^{2^{\mathrm{T}}} + \mathbf{g}$ *[80].*

**Definition 3.** *(Geometric Completeness).*
*Given a positional pair of nodes* $(x_i^t, x_j^t)$ *in a 3D graph* $\mathcal{G}$*,*
*with vectors* $a_{ij}^t \in \mathbb{R}^{1\times 3}$*,* $b_{ij}^t \in \mathbb{R}^{1\times 3}$*, and* $c_{ij}^t \in \mathbb{R}^{1\times 3}$ *derived from* $(x_i^t, x_j^t)$*,*
*a local geometric representation* $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t) \in \mathbb{R}^{3\times 3}$ *is considered*
*geometrically complete if* $\boldsymbol{\mathcal{F}}_{ij}^t$ *is non-degenerate, thereby forming*
*a local orthonormal basis located at the tangent space of* $x_i^t$ *[77].*

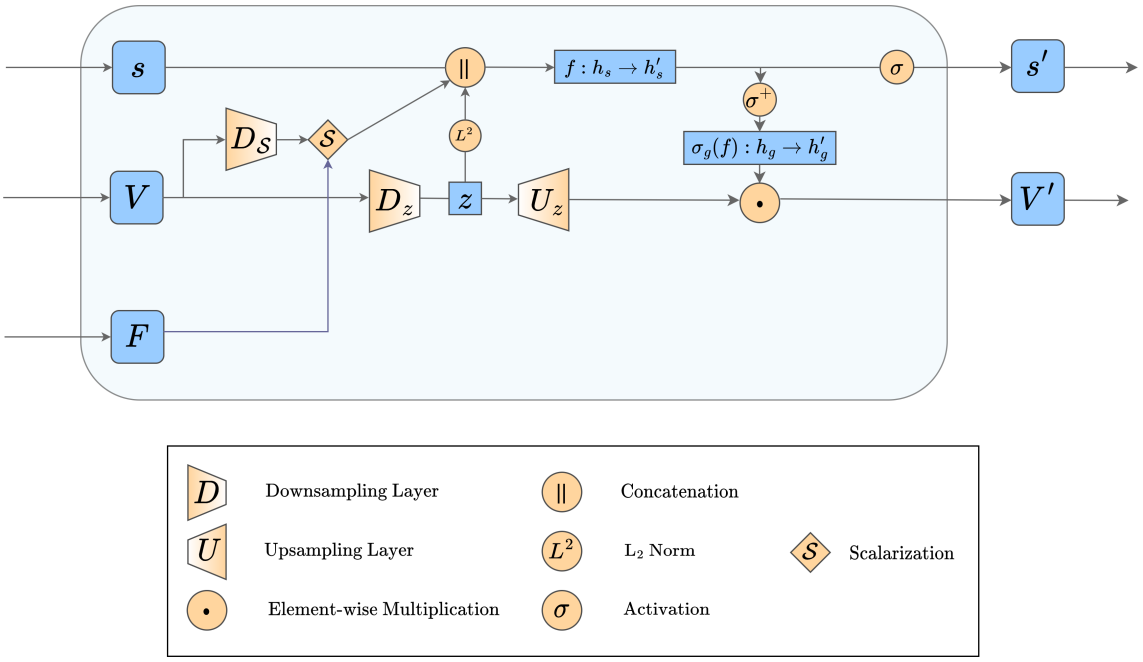



Figure 3.2: An overview of our proposed Geometry-Complete Perceptron (**GCP**) module. The **GCP** module introduces node and edge-centric encodings of 3D frames as input features that are used to directly update both scalar and vector-valued features with geometric information-completeness guarantees as well as chirality sensitivity.

### 3.3.2 GCPNet model architecture

To satisfy the geometric constraints described in Section 3.3.1, we introduce our architecture for $\mathbf{\Phi}$ satisfying Defs. (1), (2), and (3) which we refer to as the Geometry-Complete SE(3)-Equivariant Perceptron Network (GCPNet). We illustrate the GCPNet algorithm in Figure 3.1 and outline it in Algorithm 1. Subsequently, we expand on our definition for **GCP** and **GCPConv** in Section 3.3.2 in the main text and Appendix B.1, respectively, while further illustrating **GCP** in Figure 3.2.

We can then prove the following three propositions (see Appendix B.2.1 for a more detailed description of the GCPNet algorithm and its corresponding property proofs).

- **Proposition 1.** GCPNets *are SE(3)-equivariant*

  $\longrightarrow$ *Def. (1).*

- **Proposition 2.** GCPNets *are geometry self-consistent*
  $\longrightarrow$ *Def. (2).*

- **Proposition 3.** GCPNets *are geometry-complete*
  $\longrightarrow$ *Def. (3).*

### Geometry-complete perceptron module

As illustrated in Figure 3.2, GCPNet represents the features for nodes within an input graph as a tuple $(h, \chi)$ to distinguish scalar features ($h \in \mathbb{R}^h$) from vector-valued features ($\chi \in \mathbb{R}^{m \times 3}$). Similarly, GCPNet represents an input graph's edge features as a tuple $(e, \xi)$ to differentiate scalar features ($e \in \mathbb{R}^e$) from vector-valued features ($\xi \in \mathbb{R}^{x \times 3}$). For conciseness, we will subsequently refer to both node and edge feature tuples as $(s, V)$. We then define $\mathbf{GCP}_{\mathcal{F}_{ij},\lambda}(\cdot)$ to represent the **GCP** encoding process, where $\lambda$ represents a downscaling hyperparameter (e.g., 3) and $\boldsymbol{\mathcal{F}}_{ij} \in \mathbb{R}^{3 \times 3}$ denotes the SO(3)-equivariant (i.e., 3D rotation-equivariant) frames constructed using the **Localize** operation (i.e., the **EquiFrame** operation of [77]) in Algorithm 1. Specifically, the frame encodings are defined as $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t)$, with $a_{ij}^t = \frac{x_i^t - x_j^t}{\|x_i^t - x_j^t\|}, b_{ij}^t = \frac{x_i^t \times x_j^t}{\|x_i^t \times x_j^t\|}$, and $c_{ij}^t = a_{ij}^t \times b_{ij}^t$, respectively. In Appendix B, we discuss how these frame encodings are direction information-complete for edges, allowing networks incorporating them to effectively detect and leverage for downstream tasks the force fields present within real-world many-body systems such as small molecules and proteins.

*Expressing vector representations with $V$.* The **GCP** module then expresses vector representations $V$ as follows. The features $V$ with representation depth $r$ are downscaled by $\lambda$.

$$z = \{v \boldsymbol{w}_{d_z} | \boldsymbol{w}_{d_z} \in \mathbb{R}^{r \times (r/\lambda)}\} \tag{3.1}$$

Additionally, $V$ is separately downscaled in preparation to be subsequently embedded

as direction-sensitive edge scalar features.

$$V_s = \{v\boldsymbol{w}_{d_s} | \boldsymbol{w}_{d_s} \in \mathbb{R}^{r\times(3\times3)}\} \tag{3.2}$$

*Deriving scalar representations $s'$.* To update scalar representations, the **GCP** module, in the following manner, derives two invariant sources of information from $V$ and combines them with $s$:

$$q_{ij} = (V_s \cdot \mathcal{F}_{ij}) \in \mathbb{R}^9 \tag{3.3}$$

$$q = \begin{cases} \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} q_{ij} & \text{if } V_s \text{ represents nodes} \\ q_{ij} & \text{if } V_s \text{ represents edges} \end{cases} \tag{3.4}$$

$$s_{(s,q,z)} = s \cup q \cup \|z\|_2 \tag{3.5}$$

where $\cdot$ denotes the inner product, $\mathcal{N}(\cdot)$ represents the neighbors of a node, and $\|\cdot\|_2$ denotes the $L_2$ norm. Then, denote $t$ as the representation depth of $s$, and let $s_{(s,q,z)} \in \mathbb{R}^{t+9+(r/\lambda)}$ with representation depth $(t + 9 + (r/\lambda))$ be projected to $s'$ with representation depth $t'$:

$$s_v = \{s_{(s,q,z)}\boldsymbol{w}_s + \boldsymbol{b}_s | \boldsymbol{w}_s \in \mathbb{R}^{(t+9+(r/\lambda))\times t'}\} \tag{3.6}$$

$$s' = \sigma_s(s_v) \tag{3.7}$$

Note that embedding geometric frames $\mathcal{F}_{ij}$ as $q_{ij}$ in Equation 3.3 ultimately enables GCPNet to iteratively learn chirality-sensitive and global force-aware representations of each 3D network input. Moreover, Equation 3.4 allows GCPNet to encode local geometric substructures for each node, where the theoretical importance of such network behavior is discussed in detail by [77].

*Deriving vector representations $V'$.* The **GCP** module then concludes by updating

vector representations as follows:

$$V_u = \{z\boldsymbol{w}_{u_z} | \boldsymbol{w}_{u_z} \in \mathbb{R}^{(r/\lambda)\times r'}\} \tag{3.8}$$

$$V' = \{V_u \odot \sigma_g(\sigma^+(s_v)\boldsymbol{w}_g + \boldsymbol{b}_g) | \boldsymbol{w}_g \in \mathbb{R}^{t'\times r'}\} \tag{3.9}$$

where $\odot$ represents element-wise multiplication and the gating function $\sigma_g$ is applied row-wise to preserve SO(3) equivariance within $V'$.

Conceptually, the **GCP** module is autoregressively applied to tuples $(s, V)$ a total of $\omega$ times to derive rich scalar and vector-valued features. The module does so by blending both feature types iteratively with the 3D direction and information completeness guarantees provided by geometric frame encodings $\boldsymbol{\mathcal{F}}_{ij}$. We note that this model design runs in contrast with prior graph neural networks for physical systems such as GVP-GNNs [17, 76] and ClofNet [77], which are either insensitive to chemical chirality and global atomic forces or do not directly learn geometric features for downstream prediction tasks, making the proposed **GCP** module well suited for learning directly from 3D molecular graphs.

### 3.3.3 Learning from 3D graphs with GCPNet

In this section, we propose a flexible manner in which to perform 3D graph convolution with our proposed **GCP** module, as illustrated in Figure 3.1 and employed in Algorithm 1. For interested readers, in Appendix B, we provide an expanded derivation and description of how to perform 3D graph convolution with GCPNet.

#### The GCPNet algorithm

In this section, we describe our overall 3D graph convolution learning algorithm driven by GCPNet (Algorithm 1). We also discuss the rationale behind our design decisions for GCPNet and provide examples of use cases in which one might apply GCPNet

**Algorithm 1** GCPNet

**Require:** $(h_i \in \mathbf{H},\, \chi_i \in \boldsymbol{\chi})$, $(e_{ij} \in \mathbf{E},\, \xi_{ij} \in \boldsymbol{\xi})$, $x_i \in \mathbf{X}$, graph $\mathcal{G}$

1: Initialize $\mathbf{X}^0 = \mathbf{X}^C \leftarrow \textbf{Centralize}(\mathbf{X})$
2: $\boldsymbol{\mathcal{F}}_{ij} = \textbf{Localize}(x_i \in \mathbf{X}^0, x_j \in \mathbf{X}^0)$
3: Project $(h_i^0,\, \chi_i^0)$, $(e_{ij}^0,\, \xi_{ij}^0) \leftarrow \textbf{GCP}_e((h_i,\, \chi_i),\, (e_{ij},\, \xi_{ij}),\, \mathcal{F}_{ij})$
4: **for** $l = 1$ **to** $L$ **do**
5: $\quad (h_i^l,\, \chi_i^l),\, x_i^l = \textbf{GCPConv}^l((h_i^{l-1},\, \chi_i^{l-1}),\, (e_{ij}^0,\, \xi_{ij}^0),\, x_i^{l-1},\, \mathcal{F}_{ij})$
6: **if** Updating Node Positions **then**
7: $\quad \boldsymbol{\mathcal{F}}_{ij}^L = \textbf{Localize}(x_i \in \mathbf{X}^l,\, x_j \in \mathbf{X}^l)$
8: $\quad$ Finalize $(\mathbf{X}^L) \leftarrow \textbf{Decentralize}(\mathbf{X}^l)$
9: **else**
10: $\quad x_i^L = x_i^0$
11: Project $(h_i^L,\, \chi_i^L)$, $(e_{ij}^L,\, \xi_{ij}^L) \leftarrow \textbf{GCP}_p((h_i^l,\, \chi_i^l),\, (e_{ij}^0,\, \xi_{ij}^0),\, \mathcal{F}_{ij}^L)$

**Ensure:** $(h_i^L,\, \chi_i^L)$, $(e_{ij}^L,\, \xi_{ij}^L)$, $x_i^L$

for specific learning tasks.

On Line 2 of Algorithm 1, the **Centralize** operation removes the center of mass from each node position in the input graph to ensure that such positions are subsequently 3D translation-invariant.

Thereafter, following [77], the **Localize** operation on Line 3 crafts translation-invariant and SO(3)-equivariant frame encodings $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t)$. As described in more detail in Appendix B, these frame encodings are chirality-sensitive and direction information-complete for edges, imbuing networks that incorporate them with the ability to more easily detect force field interactions present in many real-world atomic systems, as we demonstrate through corresponding experiments in Section 3.4.

Before applying any geometry-complete graph convolution layers, on Line 4 we use $\textbf{GCP}_e$ to embed our input node and edge features into scalar and vector-valued values, respectively, while incorporating geometric frame information. Subsequently, in Lines 5-6, each layer of geometry-complete graph convolution is performed autoregressively via $\textbf{GCPConv}^l$ starting from these initial node and edge feature embeddings, all while maintaining information flow originating from the geometric frames $\boldsymbol{\mathcal{F}}_{ij}$.

On Lines 8 through 12, we finalize our procedure with which to update in an SE(3)-

equivariant manner the position of each node in an input 3D graph. In particular, we update node positions by residually adding learned vector-valued node features ($\chi_{v_i}^l$) to the node positions produced by the previous **GCPConv** layer ($l-1$). As shown in Appendix B, such updates are initially SO(3)-equivariant, and on Line 10 we ensure these updates also become 3D translation-equivariant by adding back to each node position the input graph's original center of mass via the **Decentralize** operation. In total, this procedure produces SE(3)-equivariant updates to node positions. Additionally, for models that update node positions, we note that Line 9 updates frame encodings $\boldsymbol{\mathcal{F}}_{ij}$ using the model's final predictions for node positions to provide more information-rich feature projections on Line 14 via $\mathbf{GCP}_p$ to conclude the forward pass of GCPNet.

**Network utilities.**

In summary, GCPNet receives an input 3D graph $\mathcal{G}$ with node positions $\boldsymbol{x}$, scalar node and edge features, $h$ and $e$, as well as vector-valued node and edge features, $\chi$ and $\xi$. The model is then capable of e.g., (1) predicting scalar node, edge, or graph-level properties while maintaining SE(3) invariance; (2) estimating vector-valued node, edge, or graph-level properties while ensuring SE(3) equivariance; or (3) updating node positions in an SE(3)-equivariant manner.

## 3.4 RESULTS

In this work, we consider four distinct modeling tasks comprised of seven datasets in total, where implementation details are discussed in Appendix B.3.

### 3.4.1 Molecular chirality detection

**Assessing model sensitivity to molecular chirality.** Molecular chirality is an essential geometric property of 3D molecules for models to consider when making pre-

dictions for downstream tasks. Simply put, this property describes the "handedness" of 3D molecules, in that, certain molecules cannot be geometrically superimposed upon a mirror reflection of themselves using only 3D rotation and translation operations. This subsequently poses a key challenge for machine learning models: Can such predictive models effectively sensitize their predictions to the effects of molecular chirality such that, under 3D reflections, their molecular feature representations change accordingly? To answer this question using modern machine learning methods, we adopt the rectus/sinister (RS) 3D molecular dataset of [81] (i.e., a 70/15/15 train/validation/test split of PubChem3D [82] where conformers correspond to the same 2D graphs in the same partition to prevent data leakage between splits) to evaluate the ability of state-of-the-art machine learning methods to distinguish between right-handed and left-handed versions of a 3D molecule. In addition, we carefully follow their experimental setup including dataset splitting; evaluation criteria; scalar feature sets of atom types, degrees, charges, numbers of hydrogens, hybridizations, and bond types and distances; and vector feature sets of atom orientations and pairwise bond displacements, respectively), where we evaluate each method's classification accuracy in distinguishing between right and left-handed versions of a molecule. Baseline methods for this task include state-of-the-art invariant neural networks (INNs) and equivariant neural networks (ENNs), where we list each method's latest results for this task as reported in [83].

**Contribution of frame embeddings for chirality sensitivity.** Table 3.1 shows that GCPNet is more accurately able to detect the effects of molecular chirality compared to all other baseline methods (including all other SE(3)-equivariant models), even without performing any hyperparameter tuning. In particular, GCPNet outperforms ChIRo [81], a GNN specifically designed to detect different forms of chirality in 3D molecules. Moreover, when we ablate GCPNet's embeddings of local geometric frames, we find that this E(3)-equivariant (i.e., scalar-wise 3D ro-

Table 3.1: Comparison of GCPNet with baseline methods for the RS task. The results are averaged over three independent runs. The top-1 (best) results for this task are in **bold**, and the second-best results are underlined.

| Type | Method | Symmetries | R/S Accuracy (%) ↑ |
|---|---|---|---|
| INN | ChIRo [83] | SE(3) | <u>98.5</u> |
| | SchNet [83] | E(3) | 54.4 |
| | DimeNet++ [83] | E(3) | 65.7 |
| | SphereNet [83] | SE(3) | 98.2 |
| ENN | EGNN [83] | E(3) | 50.4 |
| | SEGNN [83] | SE(3) | 83.4 |
| Ours | GCPNet w/o Frames | E(3) | $50.2 \pm 0.6$ |
| | GCPNet | SE(3) | $\mathbf{98.7 \pm 0.1}$ |

tation *and* reflection-invariant) version of GCPNet is no longer able to solve this important molecular recognition task, resulting in prediction accuracies at parity with random guessing. These two previous observations highlight that (1) GCPNet's local frame embeddings are critical components of the model's sensitivity to molecular chirality and that, (2) using such frame embeddings, GCPNet can flexibly learn representations of 3D molecules that are more predictive of chemical chirality compared to hand-crafted methods for such tasks. Moreover, these results highlight that, in order to effectively account for the effects of chirality on molecular structures, a method must be SE(3)-equivariant such that it employs SE(3)-invariant (and, thereby, reflection-varying) features for its scalar downstream predictions.

### 3.4.2 Protein-ligand binding affinity prediction

**Evaluating predictions of protein-ligand binding affinity.** Protein-ligand binding affinity (LBA) prediction challenges methods to estimate the binding affinity of

a protein-ligand complex as a single scalar value [84]. Accurately estimating such values in a matter of seconds using a machine learning model can provide invaluable and timely information in the typical drug discovery pipeline [85]. The corresponding dataset for this SE(3)-invariant task is derived from the ATOM3D dataset [84] and is comprised of 4,463 nonredundant protein-ligand complexes, where cross-validation splits are derived using a strict 30% sequence identity cutoff. Results are reported in terms of the root mean squared error (RMSE), Pearson's correlation ($p$), and Spearman's correlation ($Sp$) between a method's predictions on the test dataset and the corresponding ground-truth binding affinity values represented as $pK = -\log_{10}(K)$, where $K$ is the binding affinity measured in Molar units. Baseline comparison methods for this task include a variety of state-of-the-art CNNs, recurrent neural networks (RNNs), GNNs, and ENNs, with additional baselines utilizing explicit protein-ligand interaction information listed in Table 2 of the supplementary materials. Using the same dataset and dataset splits, results for these methods are reported as in [79], [86], and [87], respectively. Note, however, that due to their lack of official publicly-available PyTorch Geometric [88] source code, for this task we include simple PyTorch Geometric reproductions of PaiNN [89] and the Equivariant Transformer (ET) [90] as additional equivariant graph neural network and Transformer baselines, respectively. Consequently, due to computational resource constraints, we do not perform any hyperparameter tuning for these two methods.

The results shown in Table 3.2 reveal that, in operating on atom-level protein-ligand graph representations, GCPNet achieves the best performance for predicting protein-ligand binding affinity by a significant margin, notably improving performance across all metrics by 7% on average. Here, to the best of our knowledge, GCPNet is one of the first methods capable of achieving Pearson and Spearman binding affinity correlations greater than 0.6 on the PDBBind dataset [91] curated as part of the ATOM3D benchmark (which employs a strict 30% sequence identity cut-

off) [84]. Moreover, we find that these correlations are highly statistically significant (i.e., Pearson's p-value of $2e-50$, Spearman's p-value of $2e-49$, and Kendall's tau correlation of 0.432 with a p-value of $3e-45$).

**Ablating network components reveals impact of model design.** Denoted as "GCPNet w/o ..." in Table 3.2, our ablation studies with GCPNet for the LBA task demonstrate the contribution of each component in its model design. In particular, our proposed local frame embeddings improve GCPNet's performance by more than 15% across all metrics (GCPNet w/o Frames), where we hypothesize these performance improvements come from using these frame embeddings to enhance the model's sensitivity to molecular chirality. Similarly, our proposed residual GCP module (i.e., ResGCP) improves GCPNet's performance by 23% on average.

Specifically of interest is the observation that independent removal of scalar and vector-valued features within GCPNet appears to severely decrease GCPNet's performance for LBA prediction. Notably, removing the model's access to scalar-valued node and edge features (i.e., one-hot atom types and edge distance embeddings, respectively) degrades performance by 70% on average, while not allowing the model to access vector-valued node and edge features (i.e., sequence-based orientation vectors and pairwise atom displacement vectors, respectively) reduces performance by 42% on average. One possible explanation for these observations is that both types of feature representations the baseline GCPNet model learns (i.e., scalars and vectors) are useful for understanding protein-ligand interactions. In addition, our ablation results in Table 3.2 suggest that our proposed frame embeddings and ResGCP module are complementary to these scalar and vector-valued features in the context of predicting the binding affinity of a protein-ligand complex.

### 3.4.3 Protein model quality assessment

**Evaluating ranking predictions for protein structure decoys.** Protein structure ranking (PSR) requires methods to predict the overall quality of a 3D protein structure when comparing it to a reference (i.e., native) protein structure [84]. The quality of a protein structure is reported as a single scalar value representing a method's predicted global distance test (GDT_TS) score [92] between the provided decoy structure and the native structure. Such information is crucial in drug discovery efforts when one is tasked with designing a drug (e.g., ligand) that should bind to a particular protein target, notably when such targets have not yet had their 3D structures experimentally determined and have rather had them predicted computationally using methods such as AlphaFold 2 [15]. The respective dataset for this SE(3)-invariant task is also derived from the ATOM3D dataset [84] and is comprised of 40,950 decoy structures corresponding to 649 total targets, where cross-validation splits are created according to a target's release year in the Critical Assessment of Techniques for Protein Structure Prediction (CASP) competition [93]. Results are reported in terms of the Pearson's correlation ($p$), Spearman's correlation ($Sp$), and Kendall's tau correlation ($K$) between a method's predictions on the test dataset and the corresponding ground-truth GDT_TS values, where local results are averaged across predictions for individual targets and global results are averaged directly across all targets. Baseline comparison methods for this task include a composition of state-of-the-art CNNs, GNNs, and ENNs (including our reproductions of PaiNN and ET), as well as previous statistics-based methods. Using the same dataset and dataset splits, results for these methods are reported as in [86] and [84], respectively.

Conveying a similar message to that in Table 3.2, the results in Table 3.3 demonstrate that, in operating on atom-level protein graphs, GCPNet performs best against all other state-of-the-art models for the task of estimating a 3D protein structure's quality (i.e., PSR). In this setting, GCPNet outperforms all other methods

across all local and global metrics by 2.5% on average. Once again, GCPNet's predictions are highly statistically significant, this time with Pearson, Spearman, and Kendall tau p-values all below $1e-50$, respectively.

**Identifying components for effective protein structure ranking.** Our ablation studies with GCPNet, in the context of PSR, once more reveal that the design of our local frames, ResGCP module, and scalar and vector feature channels are all beneficial for enhancing GCPNet's ability to analyze a given 3D graph input. Here, in sensitizing the model to chemical chirality, our local frame embeddings improve GCPNet's performance for PSR by 4% on average. Similarly, our ResGCP module improves the model's performance by 5%. Interestingly, without access to scalar-valued node and edge features (i.e., the same as those used for the LBA task), GCPNet is unable to produce valid predictions for the PSR test dataset due to what appears to be a phenomenon of vector-wise latent variable collapse [94]. This finding suggests that, for the PSR task, the baseline GCPNet model relies strongly on the scalar-valued representations it produces. Lastly, including vector-valued node and edge features (i.e., the same as those used for the LBA task) within GCPNet improves the model's performance for the PSR task by 9%.

### 3.4.4 Future position forecasting for Newtonian particle systems

**Evaluating trajectory predictions for Newtonian many-body systems.** Newtonian many-body systems modeling (NMS) asks methods to forecast the future positions of particles in many-body systems of various sizes [77], bridging the gap between the domains of machine learning and physics. In our experimental results for the NMS task, the four systems (i.e., datasets) on which we evaluate each method are comprised of increasingly more nodes and are influenced by force fields of increasingly complex directional origins for which to model, namely electrostatic force fields for 5-body (ES(5)) and 20-body (ES(20)) systems as well as for 20-body systems under

the influence of an additional gravity field (G+ES(20)) and Lorentz-like force field (L+ES(20)), respectively. The four datasets for this SE(3)-equivariant task were generated using the descriptions and source code of [77], where each dataset is comprised of 7,000 total trajectories. Results are reported in terms of the mean squared error (MSE) between a method's node position predictions on the test dataset and the corresponding ground-truth node positions after 1,000 timesteps. Baseline comparison methods for this task include a collection of state-of-the-art GNNs, ENNs, and Transformers (including our reproductions of PaiNN and ET), where we list each method's latest results for this task as reported in [77].

The results in Table 3.4 show that GCPNet achieves the lowest MSE averaged across all four NMS datasets, improving upon the state-of-the-art MSE for trajectory predictions in this task by 19% on average. In particular, GCPNet achieves the best results for two of the four NMS datasets considered in this work, where these two datasets are respectively the first and third most difficult NMS datasets for methods to model. On the two remaining datasets, GCPNet matches the performance of prior state-of-the-art methods such as ClofNet [77]. Moreover, across all four datasets, GCPNet's trajectory predictions yield an RMSE of 0.0963 and achieve Pearson, Spearman, and Kendall's tau correlations of 0.999, 0.999, and 0.981, respectively, where all such correlation values are highly statistically significant (i.e., p-values $< 1e-50$). Note that, to calculate these correlation values, we score GCPNet's vector-valued predictions independently for each coordinate axis and then average the resulting metrics. Also note that we only compare methods such as ClofNet to GCPNet in the context of the NMS task, as e.g., ClofNet is specifically designed always to predict new 3D coordinates for each of its 3D graph inputs, with coordinate updates being the primary predictive target for the NMS dataset but with other tasks not targeting updated coordinates.

**Analyzing components for successful trajectory forecasting.** Once again,

our ablation studies with GCPNet demonstrate the importance of GCPNet's local frame embeddings, scalar node and edge features (i.e., invariant velocity encodings and edge type and distance embeddings, respectively), and ResGCP module. Here, we note that we were not able to include an ablation study on GCPNet's vector-valued node and edge features (i.e., velocity and orientation encodings as well as pairwise atom displacements, respectively) since they are directly used to predict node position displacements for trajectory forecasting. Table 3.4 shows that each model component synergistically enables GCPNet to achieve new state-of-the-art results for the NMS task. In enabling the model to detect global forces, our proposed local frame embeddings improve GCPNet's ability to learn many-body system dynamics by 6% on average across all dataset contexts. Specifically interesting to note is that these local frame embeddings improve the model's trajectory predictions within the most complex dataset context (i.e., L+ES(20)) by 14%, suggesting that such frame embeddings improve GCPNet's ability to learn many-body system dynamics even in the presence of complex global force fields. Furthermore, GCPNet's ResGCP module and scalar-valued features improve the model's performance for modeling many-body systems by 35% and 57%, respectively.

Across all tasks studied in this work, GCPNet improves upon the overall performance of all previous methods. Our experiments demonstrate this for both node-level (e.g., NMS) and graph-level (e.g., LBA) prediction tasks, verifying GCPNet's ability to encode useful information for both scales of granularity. Furthermore, we have demonstrated the importance of each model component within GCPNet, showing how these components are complementary to each other in the context of representation learning over 3D molecular data. Lastly, in Appendix Table B.7, we report the run time of GCPNet on each task's test dataset to enable future methods to directly compare their computational run time to that of GCPNet.

## 3.5 DISCUSSION

In this chapter, we introduced GCPNet, a state-of-the-art GNN for 3D molecular graph representation learning. We have demonstrated its utility through several benchmark studies. In the next chapter, we detail extensions of GCPNet that increase its geometric expressiveness as well as explore its applications for generative modeling of 3D molecules.

Table 3.2: Comparison of GCPNet with baseline methods for the LBA task. The results are averaged over three independent runs. The top-1 (best) results for this task are in **bold**, and the second-best results are underlined.

| Type | Method | RMSE ↓ | $p$ ↑ | $Sp$ ↑ |
|---|---|---|---|---|
| CNN | 3DCNN [79] | 1.416 ± 0.021 | 0.550 | 0.553 |
| | DeepDTA [79] | 1.866 ± 0.080 | 0.472 | 0.471 |
| | DeepAffinity [86] | 1.893 ± 0.650 | 0.415 | 0.426 |
| RNN | Bepler and Berger [79] | 1.985 ± 0.006 | 0.165 | 0.152 |
| | TAPE [79] | 1.890 ± 0.035 | 0.338 | 0.286 |
| | ProtTrans [79] | 1.544 ± 0.015 | 0.438 | 0.434 |
| GNN | GCN [79] | 1.601 ± 0.048 | 0.545 | 0.533 |
| | DGAT [86] | 1.719 ± 0.047 | 0.464 | 0.472 |
| | DGIN [86] | 1.765 ± 0.076 | 0.426 | 0.432 |
| | DGAT-GCN [86] | 1.550 ± 0.017 | 0.498 | 0.496 |
| | MaSIF [79] | 1.484 ± 0.018 | 0.467 | 0.455 |
| | IEConv [79] | 1.554 ± 0.016 | 0.414 | 0.428 |
| | Holoprot-Full Surface [79] | 1.464 ± 0.006 | 0.509 | 0.500 |
| | Holoprot-Superpixel [79] | 1.491 ± 0.004 | 0.491 | 0.482 |
| | ProNet-Amino-Acid [79] | 1.455 ± 0.009 | 0.536 | 0.526 |
| | ProNet-Backbone [79] | 1.458 ± 0.003 | 0.546 | 0.550 |
| | ProNet-All-Atom [79] | 1.463 ± 0.001 | 0.551 | 0.551 |
| | GeoSSL-DDM [87] | 1.451 ± 0.030 | <u>0.577</u> | <u>0.572</u> |
| ENN | Cormorant [86] | 1.568 ± 0.012 | 0.389 | 0.408 |
| | PaiNN | 1.698 ± 0.050 | 0.366 | 0.358 |
| | ET | 1.490 ± 0.019 | 0.564 | 0.532 |
| | GVP [86] | 1.594 ± 0.073 | 0.434 | 0.432 |
| | GBP [86] | <u>1.405</u> ± <u>0.009</u> | 0.561 | 0.557 |
| Ours | GCPNet w/o Frames | 1.485 ± 0.015 | 0.521 | 0.504 |
| | GCPNet w/o ResGCP | 1.514 ± 0.008 | 0.471 | 0.468 |
| | GCPNet w/o Scalars | 1.685 ± 0.000 | 0.050 | 0.000 |
| | GCPNet w/o Vectors | 1.727 ± 0.005 | 0.270 | 0.304 |
| | GCPNet | **1.352 ± 0.003** | **0.608** | **0.607** |

Table 3.3: Comparison of GCPNet with baseline methods for the PSR task. Local metrics are averaged across target-aggregated metrics. The best results for this task are in **bold**, and the second-best results are underlined. N/A denotes a metric that could not be computed.

| | Local | | | Global | | |
|---|---|---|---|---|---|---|
| Method | $p \uparrow$ | $Sp \uparrow$ | $K \uparrow$ | $p \uparrow$ | $Sp \uparrow$ | $K \uparrow$ |
| 3DCNN [86] | 0.557 | 0.431 | 0.308 | 0.780 | 0.789 | 0.592 |
| GCN [84] | 0.500 | 0.411 | 0.289 | 0.747 | 0.750 | 0.547 |
| ProQ3D [86] | 0.444 | 0.432 | 0.304 | 0.796 | 0.772 | 0.594 |
| VoroMQA [86] | 0.412 | 0.419 | 0.291 | 0.688 | 0.651 | 0.505 |
| RWplus [86] | 0.192 | 0.167 | 0.137 | 0.033 | 0.056 | 0.011 |
| SBROD [86] | 0.431 | 0.413 | 0.291 | 0.551 | 0.569 | 0.393 |
| Ornate [86] | 0.393 | 0.371 | 0.256 | 0.625 | 0.669 | 0.481 |
| DimeNet [86] | 0.302 | 0.351 | 0.285 | 0.614 | 0.625 | 0.431 |
| GraphQA [86] | 0.357 | 0.379 | 0.251 | 0.821 | 0.820 | 0.618 |
| PaiNN | 0.518 | 0.444 | 0.315 | 0.773 | 0.813 | 0.611 |
| ET | 0.564 | 0.466 | 0.330 | 0.813 | 0.814 | 0.611 |
| GVP [86] | 0.581 | 0.462 | 0.331 | 0.805 | 0.811 | 0.616 |
| GBP [86] | <u>0.612</u> | <u>0.517</u> | <u>0.372</u> | <u>0.856</u> | <u>0.853</u> | <u>0.656</u> |
| GCPNet w/o Frames | 0.588 | 0.512 | 0.367 | 0.854 | 0.851 | 0.657 |
| GCPNet w/o ResGCP | 0.576 | 0.509 | 0.365 | 0.852 | 0.847 | 0.648 |
| GCPNet w/o Scalars | N/A | N/A | N/A | N/A | N/A | N/A |
| GCPNet w/o Vectors | 0.571 | 0.497 | 0.356 | 0.802 | 0.804 | 0.608 |
| GCPNet | **0.616** | **0.534** | **0.385** | **0.871** | **0.869** | **0.676** |

Table 3.4: Comparison of GCPNet with baseline methods for the NMS task. Results are reported in terms of the MSE for future position prediction over four test datasets of increasing modeling difficulty, graph sizes, and composed force field complexities. The final column reports each method's MSE averaged across all four test datasets. The best results for this task are in **bold**, and the second-best results are underlined. N/A denotes an experiment that could not be performed due to a method's numerical instability.

| Method | ES(5) | ES(20) | G+ES(20) | L+ES(20) | Average |
|---|---|---|---|---|---|
| GNN [77] | 0.0131 | 0.0720 | 0.0721 | 0.0908 | 0.0620 |
| TFN [77] | 0.0236 | 0.0794 | 0.0845 | 0.1243 | 0.0780 |
| SE(3)-Transformer [77] | 0.0329 | 0.1349 | 0.1000 | 0.1438 | 0.1029 |
| Radial Field [77] | 0.0207 | 0.0377 | 0.0399 | 0.0779 | 0.0441 |
| PaiNN | 0.0158 | N/A | N/A | N/A | N/A |
| ET | 0.1653 | 0.1788 | 0.2122 | 0.2989 | 0.2138 |
| EGNN [77] | 0.0079 | 0.0128 | 0.0118 | 0.0368 | 0.0173 |
| ClofNet [77] | **0.0065** | 0.0073 | **0.0072** | 0.0251 | 0.0115 |
| GCPNet w/o Frames | 0.0067 | 0.0074 | 0.0074 | 0.0200 | 0.0103 |
| GCPNet w/o ResGCP | 0.0090 | 0.0135 | 0.0099 | 0.0278 | 0.0150 |
| GCPNet w/o Scalars | 0.0119 | 0.0173 | 0.0170 | 0.0437 | 0.0225 |
| GCPNet | 0.0070 | **0.0071** | 0.0073 | **0.0173** | **0.0097** |

# Chapter 4

# GEOMETRY-COMPLETE DIFFUSION FOR 3D MOLECULE GENERATION AND OPTIMIZATION

Adapted from Alex Morehead and Jianlin Cheng. "Geometry-complete diffusion for 3D molecule generation and optimization". *Communications Chemistry* 7.1 (2024): 150.

## 4.1 ABSTRACT

Generative deep learning methods have recently been proposed for generating 3D molecules using equivariant graph neural networks (GNNs) within a denoising diffusion framework. However, such methods are unable to learn important geometric properties of 3D molecules, as they adopt molecule-agnostic and non-geometric GNNs as their 3D graph denoising networks, which notably hinders their ability to generate valid large 3D molecules. In this chapter, we address these gaps by introducing the Geometry-Complete Diffusion Model (GCDM) for 3D molecule generation, which outperforms existing 3D molecular diffusion models by significant margins across conditional and unconditional settings for the QM9 dataset and the larger GEOM-Drugs dataset, respectively. Importantly, we demonstrate that GCDM's generative denoising process enables the model to generate a significant proportion of valid and energetically-stable large molecules at the scale of GEOM-Drugs, whereas previous methods fail to do so with the features they learn. Additionally, we show that ex-

tensions of GCDM can not only effectively design 3D molecules for specific protein pockets but can be repurposed to consistently optimize the geometry and chemical composition of existing 3D molecules for molecular stability and property specificity, demonstrating new versatility of molecular diffusion models. Code and data are freely available at `https://github.com/BioinfoMachineLearning/Bio-Diffusion`.

## 4.2 INTRODUCTION

Generative modeling has recently been experiencing a renaissance in modeling efforts driven largely by denoising diffusion probabilistic models (DDPMs). At a high level, DDPMs are trained by learning how to denoise a noisy version of an input example. For example, in the context of computer vision, Gaussian noise may be successively added to an input image with the goals of a DDPM in mind. We would then desire for a generative model of images to learn how to successfully distinguish between the original input image's feature signal and the noise added to the image thereafter. If a model can achieve such outcomes, we can use the model to generate new images by first sampling multivariate Gaussian noise and then iteratively removing, from the current state of the image, the noise predicted by the model. This classic formulation of DDPMs has achieved significant results in the space of image generation [95], audio synthesis [96], and even meta-learning by learning how to conditionally generate neural network checkpoints [97]. Furthermore, such an approach to generative modeling has expanded its reach to encompass scientific disciplines such as computational biology [98, 99, 100, 101, 102], computational chemistry [103, 104, 105], and computational physics [106].

Concurrently, the field of geometric deep learning [26] has seen a sizeable increase in research interest lately, driven largely by theoretical advances within the discipline [107] as well as by applications of such methodology [108, 72, 109, 110]. Notably, such applications even include what is considered by many researchers to be a solution to

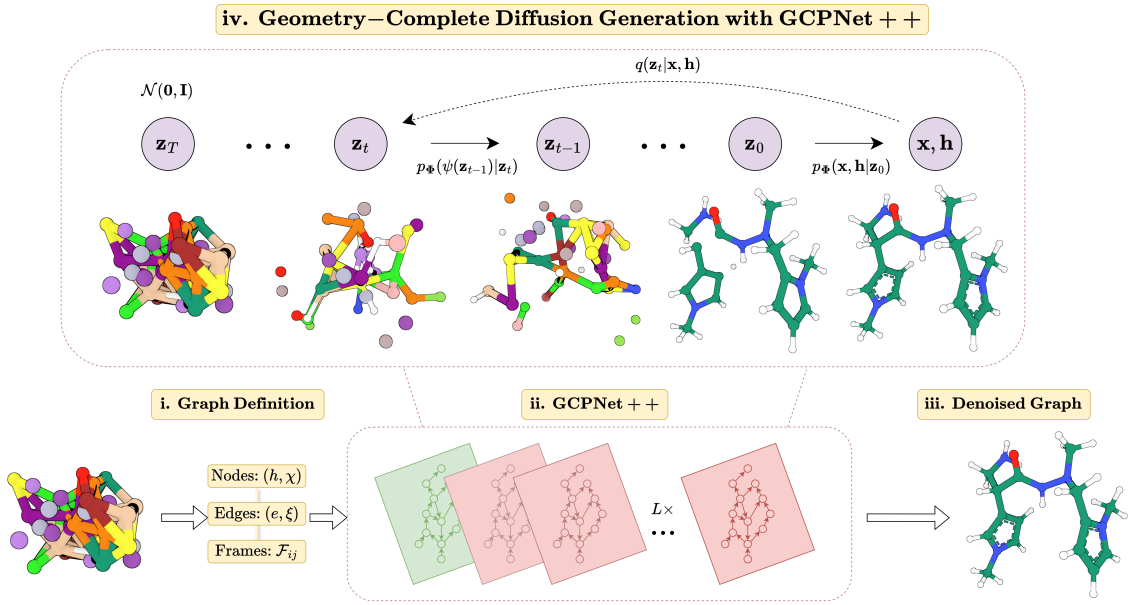



Figure 4.1: A framework overview of the proposed Geometry-Complete Diffusion Model (GCDM) for geometric and chirality-aware 3D molecule generation. The framework consists of (**i.**) a graph (topology) definition process; (**ii.**) a GCPNet-based graph neural network for SE(3)-equivariant graph representation learning; (**iii.**) denoising of 3D input graphs using GCPNet++; and (**iv.**) application of a trained GCPNet++ denoising network for 3D molecule generation.

the problem of predicting 3D protein structures from their corresponding amino acid sequences [15]. Such an outcome arose, in part, from recent advances in sequence-based language modeling efforts [11, 25] as well as from innovations in equivariant neural network modeling [111].

However, it is currently unclear how the expressiveness of geometric neural networks impacts the ability of generative methods that incorporate them to faithfully model a geometric data distribution. In addition, it is currently unknown whether diffusion models for 3D molecules can be repurposed for important, real-world tasks without retraining or fine-tuning and whether geometric diffusion models are better equipped for such tasks. Toward this end, in this work, we provide the following findings.

- Neural networks that perform message-passing with geometric quantities enable diffusion generative models of 3D molecules to generate valid and energetically-

stable large molecules whereas non-geometric message-passing networks fail to do so, where we introduce key computational metrics to enable such findings.

- Physical inductive biases such as invariant graph attention and molecular chirality both play important roles in diffusion-generating valid 3D molecules.

- Our newly-proposed Geometry-Complete Diffusion Model (GCDM - see Figure 4.1), which is the first diffusion model to incorporate the above insights and achieve the ideal type of equivariance for 3D molecule generation (i.e., SE(3) equivariance), establishes state-of-the-art (SOTA) results for conditional 3D molecule generation on the QM9 dataset as well as for unconditional molecule generation on the GEOM-Drugs dataset of large 3D molecules, for the latter more than doubling PoseBusters validity rates; generates more unique and novel small molecules for unconditional generation on the QM9 dataset; and achieves better Vina energy scores and more than twofold higher PoseBusters validity rates [112] for protein-conditioned 3D molecule generation.

- We further demonstrate that geometric diffusion models such as GCDM can consistently perform 3D molecule optimization for molecular stability as well as for specific molecular properties without requiring any retraining and can consistently do so whereas non-geometric diffusion models cannot.

## 4.3 RESULTS

### 4.3.1 Unconditional 3D molecule generation - QM9

The first dataset used in our experiments, the QM9 dataset [113], contains molecular properties and 3D atom coordinates for 130k small molecules. Each molecule in QM9 can contain up to 29 atoms after hydrogen atoms are imputed for each molecule following dataset postprocessing as in [114]. For the task of 3D molecule generation, we train GCDM to unconditionally generate molecules by producing atom types (H,

C, N, O, and F), integer atom charges, and 3D coordinates for each of the molecules' atoms. Following [115], we split QM9 into training, validation, and test partitions consisting of 100k, 18k, and 13k molecule examples, respectively.

**Metrics.** We measure each method's average negative log-likelihood (NLL) over the corresponding test dataset, for methods that report this quantity. Intuitively, a method achieving a lower test NLL compared to other methods indicates that the method can more accurately predict denoised pairings of atom types and coordinates for unseen data, implying that it has fit the underlying data distribution more precisely than other methods. In terms of molecule-specific metrics, we adopt the scoring conventions of [116] by using the distance between atom pairs and their respective atom types to predict bond types (single, double, triple, or none) for all but one baseline method (i.e., E-NF). Subsequently, we measure the proportion of generated atoms that have the right valency (atom stability - AS) and the proportion of generated molecules for which all atoms are stable (molecule stability - MS). To offer additional insights into each method's behavior for 3D molecule generation, we also report the validity (Val) of the generated molecules as determined by RDKit [117], the uniqueness of the generated molecules overall (Uniq), and whether the generated molecules pass each of the de novo chemical and structural validity tests (i.e., sanitizable, all atoms connected, valid bond lengths and angles, no internal steric clashes, flat aromatic rings and double bonds, low internal energy, correct valence, and kekulizable) proposed in the PoseBusters software suite [112] and adopted by recent works on molecule generation tasks [118, 119]. Each method's results in the top half (bottom half) of Table 4.1 are reported as the mean and standard deviation (mean and Student's t-distribution 95% confidence error intervals) ($\pm$) of each metric across three (five) test runs on QM9, respectively.

**Baselines.** Besides including a reference point for molecule quality metrics using QM9 itself (i.e., Data), we compare GCDM (a geometry-complete DDPM - i.e., GC-

DDPM) to 10 baseline models for 3D molecule generation, each trained and tested using the same corresponding QM9 splits for fair comparisons: G-Schnet [120]; Equivariant Normalizing Flows (E-NF) [116]; Graph Diffusion Models (GDM) [114] and their variations (i.e., GCM-aug); Equivariant Diffusion Models (EDM) [114]; Bridge and Bridge + Force [121]; latent diffusion models (LDMs) such as GraphLDM and its variation GraphLDM-aug [122]; as well as the state-of-the-art GeoLDM method [122]. Note that we specifically include these baselines as representative implicit bond prediction methods for which bonds are inferred using their generated molecules' atom types and inter-atom distances, in contrast to explicit bond prediction approaches such as those of [123] and [124] for fair comparisons with our method. For each of such baseline methods, we report their results as curated by [121] and [122]. We further include two GCDM ablation models to more closely analyze the impact of certain key model components within GCDM. These two ablation models include GCDM without chiral and geometry-complete local frames $\mathcal{F}_{ij}$ (i.e., GCDM w/o Frames) and GCDM without scalar message attention (SMA) applied to each edge message (i.e., GCDM w/o SMA). In Section 4.5 as well as the Supplementary Methods of Appendix C.1.2 and the Supplementary Notes of Appendix C.2, we further discuss GCDM's design, hyperparameters, and optimization with these model configurations.

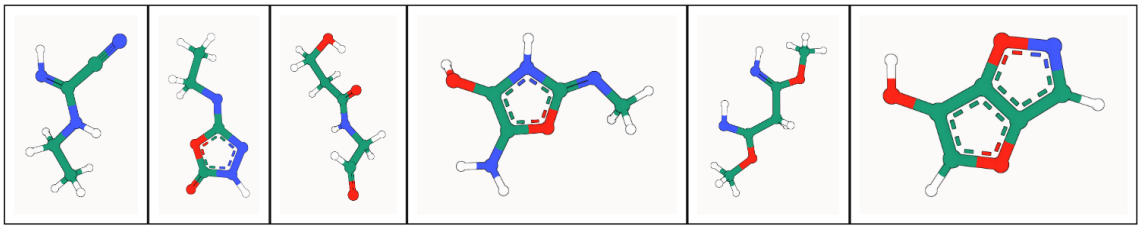



Figure 4.2: PB-valid 3D molecules generated by GCDM for the QM9 dataset. The corresponding SMILES strings for these generated small molecules, from left to right, are as follows: (a) [H]/N=C(\C#N)NCC, (b) CC[N]c1n[nH]c(=O)o1, (c) O=CCNC(=O)CCO, (d) C/N=c1/[nH]c(O)c(N)o1, (e) [H]/N=C(/C[C]([NH])OC)OC, and (f) Oc1coc2cnoc12.

| Type | Method | NLL ↓ | AS (%) ↑ | MS (%) ↑ | Val (%) ↑ | Val and Uniq (%) ↑ |
|---|---|---|---|---|---|---|
| NF | E-NF | -59.7 | 85.0 | 4.9 | 40.2 | 39.4 |
| Generative GNN | G-Schnet | - | 95.7 | 68.1 | 85.5 | 80.3 |
| DDPM | GDM | -94.7 | 97.0 | 63.2 | - | - |
| | GDM-aug | -92.5 | 97.6 | 71.6 | 90.4 | 89.5 |
| | EDM | -110.7 ± 1.5 | 98.7 ± 0.1 | 82.0 ± 0.4 | 91.9 ± 0.5 | 90.7 ± 0.6 |
| | Bridge | - | 98.7 ± 0.1 | 81.8 ± 0.2 | - | 90.2 |
| | Bridge + Force | - | 98.8 ± 0.1 | 84.6 ± 0.3 | 92.0 | 90.7 |
| LDM | GraphLDM | - | 97.2 | 70.5 | 83.6 | 82.7 |
| | GraphLDM-aug | - | 97.9 | 78.7 | 90.5 | 89.5 |
| | GeoLDM | - | **98.9** ± 0.1 | **89.4** ± 0.5 | 93.8 ± 0.4 | 92.7 ± 0.5 |
| GC-DDPM - Ours | GCDM w/o Frames | -162.3 ± 0.3 | 98.4 ± 0.0 | 81.7 ± 0.5 | 93.9 ± 0.1 | 92.7 ± 0.1 |
| | GCDM w/o SMA | -131.3 ± 0.8 | 95.7 ± 0.1 | 51.7 ± 1.4 | 83.1 ± 1.7 | 82.8 ± 1.7 |
| | GCDM | **-171.0** ± 0.2 | 98.7 ± 0.0 | 85.7 ± 0.4 | **94.8** ± 0.2 | **93.3** ± 0.0 |
| Data | | | 99.0 | 95.2 | 97.7 | 97.7 |

| Method | NLL ↓ | AS (%) ↑ | MS (%) ↑ | Val (%) ↑ | Val and Uniq (%) ↑ | Novel (%) ↑ | PB-Valid (%) ↑ |
|---|---|---|---|---|---|---|---|
| GeoLDM | - | **98.9** ± 0.0 | **89.8** ± 0.4 | 93.6 ± 0.2 | 91.8 ± 0.2 | 53.5 ± 0.6 | **93.1** ± 0.4 |
| GCDM | **-169.4** ± 0.8 | 98.7 ± 0.1 | 86.0 ± 0.7 | **94.9** ± 0.3 | **93.4** ± 0.3 | **58.7** ± 0.5 | 91.9 ± 0.5 |

Table 4.1: Comparison of GCDM with baseline methods for 3D molecule generation. The results in the top half of the table are reported in terms of the negative log-likelihood (NLL) - $\log p(\mathbf{x}, \mathbf{h}, N)$, atom stability, molecule stability, validity, and uniqueness of 10,000 samples drawn from each model, with standard deviations (±) for each model across three runs on QM9. The results in the bottom half of the table are for methods specifically evaluated across five runs on QM9 using Student's t-distribution 95% confidence intervals for per-metric errors, additionally with novelty (Novel) defined as the percentage of (valid and unique) generated molecule SMILES strings that were not found in the QM9 dataset and PoseBusters validity (PB-Valid) defined as the percentage of generated molecules that pass all relevant de novo structural and chemical sanity checks listed in Section 4.3.1. The top-1 (best) results for this task are in **bold**, and the second-best results are underlined, with - denoting a metric value that is not available.

**Results.** In the top half of Table 4.1, we see that GCDM achieves the highest percentage of probable (NLL), valid, and unique molecules compared to all baseline methods, with AS and MS results marginally lower than those of GeoLDM yet with lower standard deviations. In the bottom half of Table 4.1, where we reevaluate GCDM and GeoLDM using 5 sampling runs and report 95% confidence intervals for each metric, GCDM generates 1.6% more RDKit-valid and unique molecules and

5.2% more novel molecules compared to GeoLDM, all while offering the best reported NLL for the QM9 test dataset. This result indicates that although GeoLDM offers novelty rates close to parity (i.e., 50%), GCDM nearly matches the stability and PB-validity rates of GeoLDM while yielding novel molecules nearly 60% of the time on average, suggesting that GCDM may prove more useful for accurately exploring the space of novel yet valid small molecules. Our ablation of SMA within GCDM demonstrates that, to generate stable 3D molecules, GCDM heavily relies on both being able to perform a lightweight version of fully-connected graph self-attention [11], which suggests avenues of future research that will be required to scale up such generative models to large biomolecules such as proteins. Additionally, removing geometric local frame embeddings from GCDM reveals that the inductive biases of molecular chirality and geometry-completeness are important contributing factors in GCDM achieving these SOTA results. Figure 4.2 illustrates PoseBusters-valid examples of QM9-sized molecules generated by GCDM.

### 4.3.2 Property-conditional 3D molecule generation - QM9

**Baselines.** Towards the practical use case of conditional generation of 3D molecules, we compare GCDM to existing E(3)-equivariant models, EDM [114] and GeoLDM [122], as well as to two naive baselines: "Naive (Upper-bound)" where a molecular property classifier $\phi_c$ predicts molecular properties given a method's generated 3D molecules and shuffled (i.e., random) property labels; and "# Atoms" where one uses the numbers of atoms in a method's generated 3D molecules to predict their molecular properties. For each baseline method, we report its mean absolute error (MAE) in terms of molecular property prediction by an ensemble of three EGNN classifiers $\phi_c$ [18] as reported in [114]. For GCDM, we train each conditional model by conditioning it on one of six distinct molecular property feature inputs - $\alpha$, gap, homo, lumo, $\mu$, and $C_v$ - for approximately 1,500 epochs using the QM9 validation

split of [114] as the model's training dataset and the QM9 training split of [114] as the corresponding EGNN classifier ensemble's training dataset. Consequently, one can expect the gap between a method's performance and that of "QM9 (Lower-bound)" to decrease as the method more accurately generates property-specific molecules.

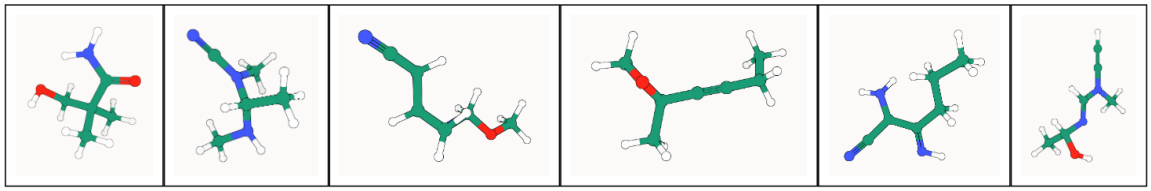
| Task<br>Units | $\alpha \downarrow$<br>$Bohr^3$ | $\Delta\epsilon \downarrow$<br>$meV$ | $\epsilon_{HOMO} \downarrow$<br>$meV$ | $\epsilon_{LUMO} \downarrow$<br>$meV$ | $\mu \downarrow$<br>$D$ | $C_v \downarrow$<br>$\frac{cal}{mol}K$ |
|---|---|---|---|---|---|---|
| Naive (Upper-bound) | 9.01 | 1470 | 645 | 1457 | 1.616 | 6.857 |
| # Atoms | 3.86 | 866 | 426 | 813 | 1.053 | 1.971 |
| EDM | 2.76 | 655 | 356 | 584 | 1.111 | 1.101 |
| GeoLDM | <u>2.37</u> | **587** | **340** | <u>522</u> | <u>1.108</u> | <u>1.025</u> |
| GCDM | **1.97** | <u>602</u> | <u>344</u> | **479** | **0.844** | **0.689** |
| QM9 (Lower-bound) | 0.10 | 64 | 39 | 36 | 0.043 | 0.040 |

| Task<br>Units | $\alpha \downarrow$<br>$Bohr^3$ | $\Delta\epsilon \downarrow$<br>$meV$ | $\epsilon_{HOMO} \downarrow$<br>$meV$ | $\epsilon_{LUMO} \downarrow$<br>$meV$ | $\mu \downarrow$<br>$D$ | $C_v \downarrow$<br>$\frac{cal}{mol}K$ |
|---|---|---|---|---|---|---|
| GeoLDM | 2.77 ± 0.12 | <u>655</u> ± 20.57 | 357 ± 5.68 | <u>565</u> ± 10.62 | <u>1.089</u> ± 0.02 | 1.070 ± 0.04 |
| GCDM | **1.99** ± 0.01 | **595** ± 14.34 | **346** ± 1.23 | **480** ± 6.58 | **0.855** ± 0.00 | **0.698** ± 0.01 |

| Metric | $\alpha$ PB-Valid (%) ↑ | $\Delta\epsilon$ PB-Valid (%) ↑ | $\epsilon_{HOMO}$ PB-Valid (%) ↑ | $\epsilon_{LUMO}$ PB-Valid (%) ↑ | $\mu$ PB-Valid (%) ↑ | $C_v$ PB-Valid (%) ↑ |
|---|---|---|---|---|---|---|
| GeoLDM | 93.7 ± 0.5 | 92.8 ± 0.3 | 93.9 ± 0.4 | 93.3 ± 0.6 | 93.2 ± 1.3 | 92.5 ± 0.8 |
| GCDM | 92.3 ± 0.3 | 92.5 ± 0.8 | 92.7 ± 0.5 | 92.7 ± 0.6 | 92.4 ± 0.4 | 91.7 ± 0.4 |

Table 4.2: Comparison of GCDM with baseline methods for property-conditional 3D molecule generation. The results in the top half of the table are reported in terms of the MAE for molecular property prediction by an EGNN classifier $\phi_c$ on a QM9 subset, with results listed for GCDM-generated samples as well as for four separate baseline methods. The results in the bottom half of the table (where GeoLDM is retrained using its official code repository due to the unavailability of its conditional model checkpoints) are likewise listed for selected methods yet instead report (across an ensemble of three separately-trained EGNN property classifier models, each with a distinct random seed) Student's t-distribution 95% confidence error intervals for each property metric as well as the percentage of PoseBusters-validated (PB-Valid) de novo generated molecules. The top-1 (best) conditioning results for this task are in **bold**, and the second-best results are <u>underlined</u>.

**Results.** We see in Table 4.2 that GCDM achieves the best overall results



(a) $\alpha = 68.7$ (b) 73.5 (c) 79.1 (d) 82.5 (e) 86.0 (f) 93.6

Figure 4.3: PB-valid 3D molecules generated by GCDM using increasing values of $\alpha$. The structural characteristics of the generated molecules are gradually altered as $\alpha$ ranges from 68.7 (a) to 93.6 (f).

compared to all baseline methods in conditioning on a given molecular property, with conditionally-generated samples shown in Figure 4.3 (Note: PSI4-computed property values [125] for (a) and (f) are 69.1 Bohr$^3$ (energy: -402 a.u.) and 89.7 Bohr$^3$ (energy: -419 a.u.), respectively, at the DFT/B3LYP/6-31G(2DF,P) level of theory [113, 126]). In particular, as shown in the bottom half of this table, GCDM surpasses the MAE results of the SOTA GeoLDM method (by 19% on average) for all six molecular properties - $\alpha$, gap, homo, lumo, $\mu$, and $C_v$ - by 28%, 9%, 3%, 15%, 21%, and 35%, respectively, while nearly matching the PB-Valid rates of GeoLDM (similar to the results in Table 4.1). These results qualitatively and quantitatively demonstrate that, using geometry-complete diffusion, GCDM enables notably precise generation of 3D molecules with specific molecular properties (e.g., $\alpha$ - polarizability).

### 4.3.3 Unconditional 3D molecule generation - GEOM-Drugs

The second dataset used in our experiments, the GEOM-Drugs dataset, is a well-known source of large, 3D molecular conformers for downstream machine learning tasks. It contains 430k molecules, each with 44 atoms on average and with up to as many as 181 atoms after hydrogen atoms are imputed for each molecule following dataset postprocessing as in [114]. For this experiment, we collect the 30 lowest-energy conformers corresponding to a molecule and task each baseline method with generating new molecules with 3D positions and types for each constituent atom.

Here, we also adopt the negative log-likelihood, atom stability, and molecule stability metrics as defined in Section 4.3.1 and train GCDM using the same hyperparameters as listed in the Supplementary Note of Appendix C.2.2, with the exception of training for approximately 75 epochs on GEOM-Drugs.

**Baselines.** In this experiment, we compare GCDM to several state-of-the-art baseline methods for 3D molecule generation on GEOM-Drugs. Similar to our experiments on QM9, in addition to including a reference point for molecule quality metrics using GEOM-Drugs itself (i.e., Data), here we also compare against E-NF, GDM, GDM-aug, EDM, Bridge along with its variant Bridge + Force, as well as GraphLDM, GraphLDM-aug, and GeoLDM. As in Section 4.3.1, each method's results in the top half (bottom half) of the table are reported as the mean and standard deviation (mean and Student's t-distribution 95% confidence interval) (±) of each metric across three (five) test runs on GEOM-Drugs.

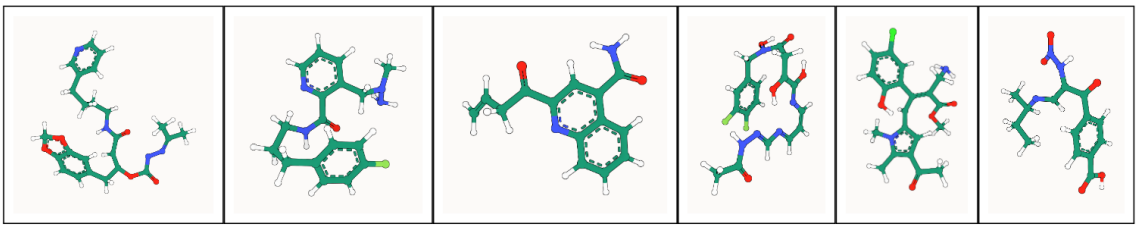



Figure 4.4: PB-valid 3D molecules generated by GCDM for the GEOM-Drugs dataset. The corresponding SMILES strings for these generated large molecules, from left to right, are as follows: (a) CC(C)=N[N]C(=O)O[C]([CH]C(=O)NCCCCc1cccnc1)Cc1ccc2c(c1)OCO2, (b) CN(N)Cc1cccnc1C(=O)NCCCc1ccc(F)cc1, (c) C=CCC(=O)c1cc(C(N)=O)c2ccccc2n1, (d) CC(=O)N/N=C/N=C/C=C\N=C(/O)[C](O)CC(=O)N(O)Cc1ccc(F)c(F)c1, (e) COC(=O)/C(CN)=C(\[CH]c1cc(C(C)=O)c(C)n1C)c1cc(Cl)ccc1O, and (f) CC[C@@H](C)/N=C/[C](N[N+](=O)[O-])C(=O)c1ccc(C(=O)O)cc1.

**Results.** To start, Table 4.3 displays an interesting phenomenon that is important to note: Due to the size and atomic complexity of GEOM-Drugs' molecules and the subsequent errors accumulated when estimating bond types based on such inter-atom distances, the baseline results for the molecule stability metrics measured here (i.e.,

| Type | Method | NLL ↓ | AS (%) ↑ | MS (%) ↑ |
|---|---|---|---|---|
| NF | E-NF | - | 75.0 | 0.0 |
| DDPM | GDM | -14.2 | 75.0 | 0.0 |
| | GDM-aug | -58.3 | 77.7 | 0.0 |
| | EDM | <u>-137.1</u> | 81.3 | 0.0 |
| | Bridge | - | 81.0 ± 0.7 | 0.0 |
| | Bridge + Force | - | 82.4 ± 0.8 | 0.0 |
| LDM | GraphLDM | - | 76.2 | 0.0 |
| | GraphLDM-aug | - | 79.6 | 0.0 |
| | GeoLDM | - | <u>84.4</u> | <u>0.0</u> |
| GC-DDPM - Ours | GCDM w/o Frames | 769.7 | 88.0 ± 0.3 | 3.4 ± 0.3 |
| | GCDM w/o SMA | 3505.5 | 43.9 ± 3.6 | 0.1 ± 0.0 |
| | GCDM | **-234.3** | **89.0** ± 0.8 | **5.2** ± 1.1 |
| Data | | | 86.5 | 2.8 |

| Method | NLL ↓ | AS (%) ↑ | MS (%) ↑ | Val (%) ↑ | Val and Uniq (%) ↑ | Novel (%) ↑ | PB-Valid (%) ↑ |
|---|---|---|---|---|---|---|---|
| GeoLDM | - | <u>84.4</u> ± 0.1 | <u>0.6</u> ± 0.1 | **99.5** ±0.1 | **99.4** ± 0.1 | - | <u>38.3</u> ± 0.5 |
| GCDM | **-215.1** ± 3.8 | **88.1** ± 0.1 | **4.3** ± 0.4 | <u>95.5</u> ± 0.1 | <u>95.5</u> ± 0.1 | **95.5** ± 0.1 | **77.0** ± 0.1 |

Table 4.3: Comparison of GCDM with baseline methods for 3D molecule generation. The results in the top half of the table are reported in terms of each method's negative log-likelihood, atom stability, and molecule stability with standard deviations (±) across three runs on GEOM-Drugs, each drawing 10,000 samples from the model. The results in the bottom half of the table are for methods specifically evaluated across five runs on QM9 using Student's t-distribution 95% confidence intervals for per-metric errors, additionally with validity and uniqueness (Val and Uniq), novelty (Novel), and PoseBusters validity (PB-Valid) defined likewise as in Section 4.3.1; The top-1 (best) results for this task are in **bold**, and the second-best results are <u>underlined</u>.

Data) are much lower than those collected for the QM9 dataset. Thus, reporting additional chemical and structural validity metrics (e.g., PB-Valid) for comparison is crucial to accurately assess a method's performance in this context, which we do

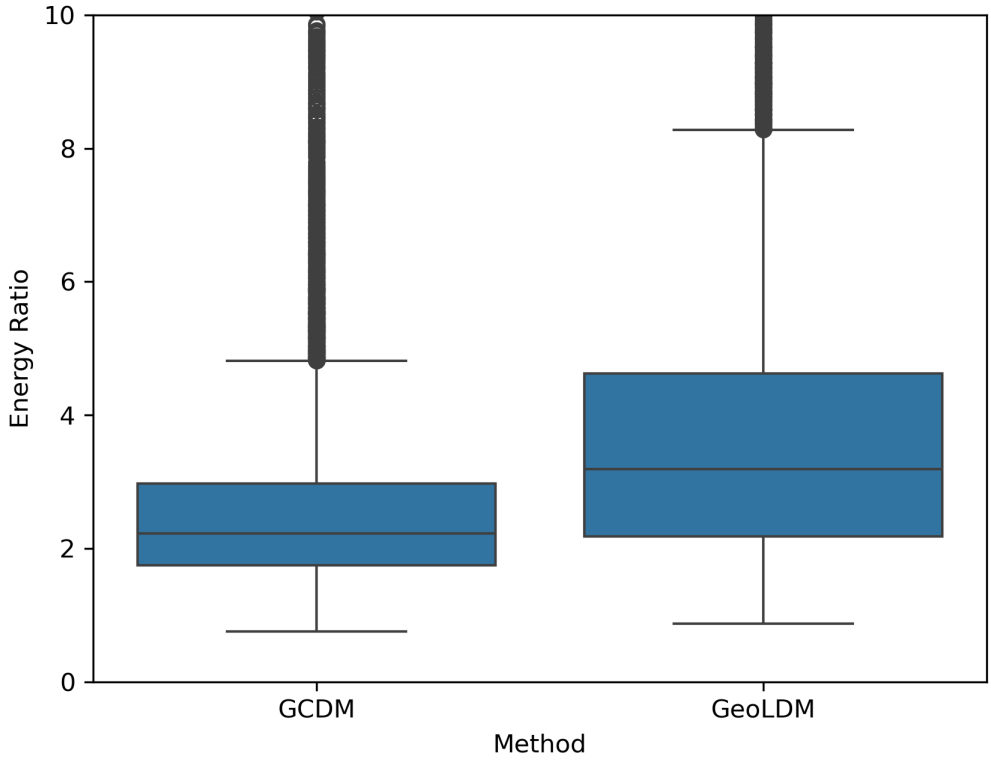



Figure 4.5: A comparison of the energy ratios [112] of 10,000 large 3D molecules generated by GCDM and GeoLDM, a baseline state-of-the-art method. Employing Student's t-distribution 95% confidence intervals, GCDM achieves a mean energy ratio of 2.98 ± 0.13, whereas GeoLDM yields a mean energy ratio of 4.19 ± 0.09.

in the bottom half of Table 4.3. Nonetheless, for GEOM-Drugs, GCDM supersedes EDM's SOTA negative log-likelihood results by 57% and advances GeoLDM's SOTA atom and molecule stability results by 4% and more than sixfold, respectively. More importantly, however, GCDM can generate a significant proportion of PB-valid large molecules, surpassing even the reference molecule stability rate of the GEOM-Drugs dataset (i.e., 2.8) by 54%, demonstrating that geometric diffusion models such as GCDM can not only effectively generate valid large molecules but can also generalize beyond the native distribution of stable molecules within GEOM-Drugs.

Figure 4.4 illustrates PoseBusters-valid examples of large molecules generated by GCDM at the scale of GEOM-Drugs. As an example of the notion that GCDM

produces low energy structures for a generated molecular graph, the free energies for Figures 4.4 (a) and (f) were computed to be -3 kcal/mol and -2 kcal/mol, respectively, using CREST 2.12 [127] at the GFN2-xTB level of theory (which matches the corresponding free energy distribution mean for the GEOM-Drugs dataset (-2.5 kcal/mol) as illustrated in Figure 2 of [128]). Lastly, to detect whether a method, in aggregate, generates molecules with unlikely 3D conformations, a generated molecule's energy ratio is defined as in [112] to be the ratio of the molecule's UFF-computed energy [129] and the mean of 50 RDKit ETKDGv3-generated conformers [130] of the same molecular graph. Note that, as discussed by [131], generated molecules with an energy ratio greater than 7 are considered to have highly unlikely 3D conformations. Subsequently, Figure 4.5 reveals that the average energy ratio of GCDM's large 3D molecules is notably lower and more tightly bounded compared to GeoLDM, the baseline SOTA method for this task, indicating that GCDM also generates more energetically-stable 3D molecule conformations compared to prior methods.

### 4.3.4 Property-guided 3D molecule optimization - QM9

To evaluate whether molecular diffusion models can not only generate new 3D molecules but can also optimize existing small molecules using molecular property guidance, we adopt the QM9 dataset for the following experiment. First, we use an unconditional GCDM model to generate 1,000 3D molecules using 10 time steps of time-scaled reverse diffusion (to leave such molecules in an unoptimized state), and then we provide these molecules to a separate property-conditional diffusion model for optimization of the molecules towards the conditional model's respective property. This conditional model accepts these 3D molecules as intermediate states for 100 and 250 time steps of property-guided optimization of the molecules' atom types and 3D coordinates. Lastly, we repurpose our experimental setup from Section 4.3.2 to score these optimized molecules using an ensemble of external property classifier models to evaluate

(1) how much the optimized molecules' predicted property values have been improved for the respective property (first metric) and (2) whether and how much the optimized molecules' stability (as defined in Section 4.3.1) has been changed during optimization (second metric).

**Baselines.** Baseline methods for this experiment include EDM [114] and GCDM, where both methods use similar experimental setups for evaluation. Our baseline methods also include property-specificity and molecule stability measures of the initial (unconditional) 3D molecules to demonstrate how much molecular diffusion models can modify or improve these existing 3D molecules in terms of how property-specific and stable they are. As in Section 4.3.2, property specificity is measured in terms of the corresponding property classifier's MAE for a given molecule with a targeted property value, reporting the mean and Student's t-distribution 95% confidence interval for each property MAE across an ensemble of three corresponding classifiers. Molecular stability (i.e., Mol Stable (%)), here abbreviated at $MS$, is defined as in Section 4.3.1.

**Results.** In this section, we quantitatively explore (in Figure 4.6) whether and how much generative models can reduce the property-specific MAE and improve the molecular stability of a batch of existing 3D molecules. In particular, Figure 4.6 showcases a practical finding: geometric diffusion models such as GCDM can effectively be repurposed as 3D molecule optimization methods with minimal modifications, improving both a molecule's stability and property specificity. This finding empirically supports the idea that molecular denoising diffusion models may be applied in the optimization stage of the typical drug discovery pipeline [132] to experiment with a wider range of potential drug candidates (post-optimization) more quickly than previously possible. Simultaneously, the baseline EDM method fails to consistently optimize the stability and property specificity of existing 3D molecules, which suggests that geometric methods such as GCDM are theoretically and empirically better suited for

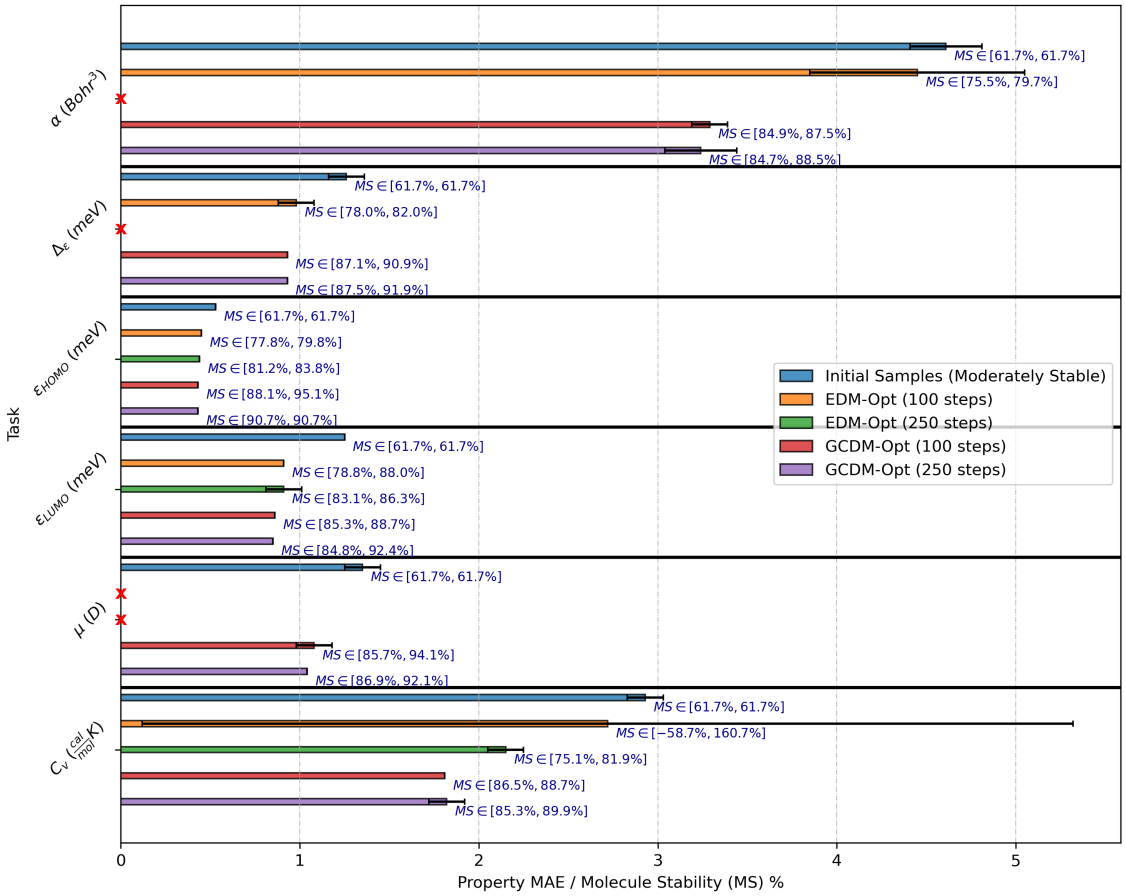



Figure 4.6: Comparison of GCDM with baseline methods for property-guided 3D molecule optimization. The results are reported in terms of molecular stability ($MS$) and the MAE for molecular property prediction by an ensemble of three EGNN classifiers $\phi_c$ (each trained on the same QM9 subset using a distinct random seed) yielding corresponding Student's t-distribution 95% confidence intervals, with results listed for EDM and GCDM-optimized samples as well as the molecule generation baseline ("Initial Samples"). Note that x denotes a missing bar representing outlier property MAEs greater than 50. Alternatively, tabular results are given in Table C.1 of the Supplementary Results of Appendix C.3.1.

such tasks. Notably, on average, with 100 time steps GCDM improves the stability of the initial molecules by over 25% and their specificity for each molecular property by over 27%, whereas for the properties it can optimize with 100 time steps, EDM improves the stability of the molecules by 13% and their property specificity by 15%. Lastly, it is worth noting that increasing the number of optimization time steps from 100 to 250 steps inconsistently leads to further improvements to molecules' stability

and property specificity, indicating that the optimization trajectory likely reaches a local minimum around 100 time steps and hence rationalizes reducing the required compute time for optimizing 1,000 molecules e.g., from 15 minutes (for 250 steps) to 5 minutes (for 100 steps).

### 4.3.5 Protein-conditional 3D molecule generation

To investigate whether geometry-complete methods can enhance the ability of molecular diffusion models to generate 3D models within a given protein pocket (i.e., to perform structure-based drug design (SBDD)), in this experiment, we adopt the standard Binding MOAD [133] and CrossDocked [134] datasets for training and evaluation of GCDM-SBDD, our geometry-complete, diffusion generative model based on GCPNet++ that extends the diffusion framework of [135] for protein pocket-aware molecule generation. The Binding MOAD dataset consists of 100,000 high-quality protein-ligand complexes for training and 130 proteins for testing, with a 30% sequence identity threshold being used to define this cross-validation split. Similarly, the CrossDocked dataset contains 40,484 high-quality protein-ligand complexes split between training (40,354) and test (100) partitions using proteins' enzyme commission numbers as described by [135].

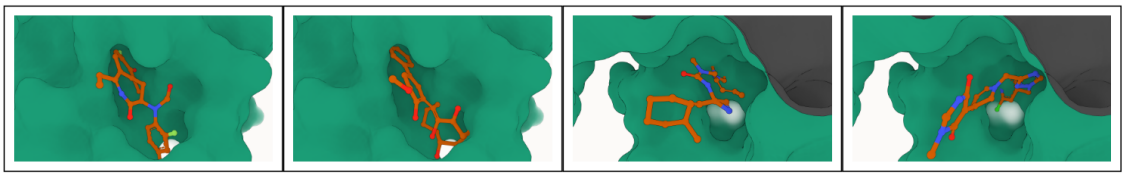
**Baselines.** Baseline methods for this experiment include DiffSBDD-cond [135] and DiffSBDD-joint [135]. We compare these methods to our proposed geometry-complete protein-aware diffusion model, GCDM-SBDD, using metrics that assess the properties, and thereby the quality, of each method's generated molecules. These molecule-averaged metrics include a method's average Vina score (computed using QuickVina 2.1) [136] as a physics-based estimate of a ligand's estimated binding affinity with a target protein, measured in units of kcal/mol (lower is better); average drug likeliness QED [137] (computed using RDKit 2022.03.2); average synthesizability [138] (computed using the procedure introduced by [139]) as an increasing measure

| Dataset | Method | Vina (kcal/mol, ↓) | QED (↑) | SA (↑) | Lipinski (↑) | Diversity (↑) | PB-Valid (%) (↑) |
|---|---|---|---|---|---|---|---|
| BM | DiffSBDD-cond (Cα) | $-5.784 \pm 0.03$ | $0.433 \pm 0.00$ | $0.616 \pm 0.00$ | $4.719 \pm 0.01$ | $0.848 \pm 0.00$ | $16.6 \pm 0.6$ / $1.7 \pm 0.2$ |
| | DiffSBDD-joint (Cα) | $-5.882 \pm 0.05$ | $0.474 \pm 0.00$ | $0.631 \pm 0.00$ | $4.835 \pm 0.01$ | $0.852 \pm 0.00$ | $10.7 \pm 0.5$ / $0.7 \pm 0.1$ |
| | GCDM-SBDD-cond (Cα) (Ours) | **-6.250** $\pm 0.03$ | $\underline{0.465} \pm 0.00$ | $\underline{0.618} \pm 0.00$ | $4.661 \pm 0.01$ | $0.806 \pm 0.00$ | **40.8** $\pm 0.8$ / **6.8** $\pm 0.4$ |
| | GCDM-SBDD-joint (Cα) (Ours) | $\underline{-6.159} \pm 0.06$ | $0.459 \pm 0.00$ | $0.584 \pm 0.00$ | $4.609 \pm 0.02$ | $0.794 \pm 0.00$ | $\underline{37.3} \pm 0.8$ / $\underline{2.0} \pm 0.2$ |
| | Reference | $-8.328 \pm 0.04$ | $0.602 \pm 0.00$ | $0.336 \pm 0.00$ | $4.838 \pm 0.01$ | – | – |
| CD | DiffSBDD-cond (Cα) | $-5.540 \pm 0.03$ | $0.449 \pm 0.00$ | $0.636 \pm 0.00$ | $4.735 \pm 0.01$ | $0.818 \pm 0.00$ | $40.7 \pm 1.0$ / $12.4 \pm 0.6$ |
| | DiffSBDD-joint (Cα) | $-5.735 \pm 0.05$ | $0.420 \pm 0.00$ | $0.662 \pm 0.00$ | $4.859 \pm 0.01$ | $0.890 \pm 0.00$ | $34.1 \pm 0.9$ / $6.2 \pm 0.5$ |
| | GCDM-SBDD-cond (Cα) (Ours) | **-5.955** $\pm 0.04$ | $\underline{0.457} \pm 0.00$ | $\underline{0.640} \pm 0.00$ | $\underline{4.758} \pm 0.02$ | $0.795 \pm 0.00$ | $38.1 \pm 1.0$ / **15.7** $\pm 0.7$ |
| | GCDM-SBDD-joint (Cα) (Ours) | $\underline{-5.870} \pm 0.03$ | **0.458** $\pm 0.00$ | $0.631 \pm 0.00$ | $4.701 \pm 0.02$ | $0.810 \pm 0.00$ | **46.8** $\pm 1.0$ / $6.5 \pm 0.5$ |
| | Reference | $-6.871 \pm 0.04$ | $0.476 \pm 0.00$ | $0.728 \pm 0.00$ | $4.340 \pm 0.00$ | – | – |

Table 4.4: Evaluation of generated molecules for target protein pockets from the Binding MOAD (BM) and CrossDocked (CD) test datasets. Our proposed method, GCDM-SBDD, achieves the best results for the metrics listed in **bold** and the second-best results for the metrics underlined. For each metric, a method's mean and Student's t-distribution 95% confidence error interval (±) is reported over 100 generated molecules for each test pocket. Additionally, the PB-Valid metric is defined as the percentage of generated molecules that pass all docking-relevant structural and chemical sanity checks proposed by [112], with the validity ratio to the left (right) of each / denoting the percentage of valid molecules without (with) consideration of protein-ligand steric clashes.

of the ease of synthesizing a given molecule (higher is better); on average how many rules of Lipinski's rule of five are satisfied by a ligand [140] (computed compositionally using RDKit 2022.03.2); and average diversity in mean pairwise Tanimoto distances [141, 142] (derived manually using fingerprints and Tanimoto similarities computed by RDKit 2022.03.2). Following established conventions for 3D molecule generation [114], the size of each ligand to generate was determined using the ligand size distribution of the respective training dataset. Note that, in this context, "joint" and "cond" configurations represent generating a molecule for a protein target, respectively, with and without also modifying the coordinates of the binding pocket within the protein target. Also note that, similar to our experiments in Sections 4.3.1 - 4.3.4, the GCDM-SBDD model uses 9 GCP message-passing layers along with 256 (64) and 32 (16) invariant (equivariant) node and edge features, respectively.

**Results.** Table 4.4 shows that, across both of the standard SBDD datasets (i.e., Binding MOAD and CrossDocked), GCDM-SBDD generates more clash-free (PB-Valid) and lower energy (Vina) molecules compared to prior methods. Moreover,



(a) 2FKY: -8.7 Vina (b) 2FKY: -9.7 Vina (c) 4Z2G: -8.2 Vina (d) 4Z2G: -8.7 Vina

Figure 4.7: GCDM-SBDD molecules generated for BM (a-b) and CD (c-d) test proteins. Vina energy scores for these selected pocket-binding molecules range from -8.2 (c) to -9.7 (b).

across all other metrics, GCDM-SBDD achieves comparable or better results in terms of drug-likeness measures (e.g., QED) and comparable results for all other molecule metrics without performing any hyperparameter tuning due to compute constraints. These results suggest that GCDM, with GCPNet++ as its denoising neural network, not only works well for de novo 3D molecule generation but also protein target-specific 3D molecule generation, notably expanding the number of real-world application areas of GCDM. Concretely, GCDM-SBDD improves upon DiffSBDD's average Vina energy scores by 8% on average across both datasets while generating more than twice as many PB-valid "candidate" molecules for the more challenging Binding MOAD dataset.

As suggested by [112], the gap between the PB-Valid ratios in Table 4.4 without and with protein-ligand steric clashes considered for both GCDM-SBDD and DiffSBDD suggests that deep learning-based drug design methods for targeted protein pockets can likely benefit significantly from interaction-aware molecular dynamics relaxation following protein-conditional molecule generation, which may allow for many generated "candidate" molecules to have their PB validity "recovered" by such relaxation. Nonetheless, Figure 4.7 demonstrates that GCDM can consistently generate clash-free realistic and diverse 3D molecules with low Vina energies for unseen protein targets.

## 4.4 DISCUSSION

While previous methods for 3D molecule generation have possessed insufficient geometric and molecular priors for scaling well to a variety of molecular datasets, in this chapter, we introduced a geometry-complete diffusion model (GCDM) that establishes a clear performance advantage over previous methods, generating more realistic, stable, valid, unique, and property-specific 3D molecules, while enabling the generation of many large 3D molecules that are energetically stable as well as chemically and structurally valid. Moreover, GCDM does so without complex modeling techniques such as latent diffusion, which suggests that GCDM's results could likely be further improved by expanding upon these techniques [122]. Although GCDM's results here are promising, since it (like previous methods) requires fully-connected graph attention as well as 1,000 time steps to generate a high-quality batch of 3D molecules, using it to generate several thousand large molecules can take a notable amount of time (e.g., 15 minutes to generate 250 new large molecules). As such, future research with GCDM could involve adding new time-efficient graph construction or sampling algorithms [143] or exploring the impact of higher-order (e.g., type-2 tensor) yet efficient geometric expressiveness [144] on 3D generative models to accelerate sample generation and increase sample quality. Furthermore, integrating additional external tools for assessing the quality and rationality of generated molecules [145] is a promising direction for future work.

## 4.5 METHODS

### 4.5.1 Problem setting

In this work, our goal is to generate new 3D molecules either unconditionally or conditioned on user-specified properties. We represent a molecular point cloud (e.g., 3D molecule) as a fully-connected 3D graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ and $\mathcal{E}$ representing

the graph's sets of nodes and edges, respectively, and $N = |\mathcal{V}|$ and $E = |\mathcal{E}|$ representing the numbers of nodes and edges in the graph, accordingly. In addition, $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N) \in \mathbb{R}^{N\times 3}$ represents the respective Cartesian coordinates for each node (i.e., atom). Each node in $\mathcal{G}$ is described by scalar features $\mathbf{H} \in \mathbb{R}^{N\times h}$ and $m$ vector-valued features $\boldsymbol{\chi} \in \mathbb{R}^{N\times(m\times 3)}$. Likewise, each edge in $\mathcal{G}$ is described by scalar features $\mathbf{E} \in \mathbb{R}^{E\times e}$ and $x$ vector-valued features $\boldsymbol{\xi} \in \mathbb{R}^{E\times(x\times 3)}$. Then, let $\mathcal{M} = [\mathbf{X}, \mathbf{H}]$ represent the molecules (i.e., atom coordinates and atom types) our method is tasked with generating, where $[\cdot, \cdot]$ denotes the concatenation of two variables. Important to note is that the input features $\mathbf{H}$ and $\mathbf{E}$ are invariant to 3D roto-translations, whereas the input vector features $\mathbf{X}$, $\boldsymbol{\chi}$ and $\boldsymbol{\xi}$ are equivariant to 3D roto-translations. Lastly, in particular, we design a denoising neural network $\boldsymbol{\Phi}$ to be equivariant to 3D roto-translations (i.e., SE(3)-equivariant) by defining it such that its internal operations and outputs match corresponding 3D roto-translations acting upon its inputs.

### 4.5.2 Overview of GCDM

We will now introduce GCDM, a new Geometry-Complete SE(3)-Equivariant Diffusion Model. GCDM defines a joint noising process on equivariant atom coordinates $\mathbf{x}$ and invariant atom types $\mathbf{h}$ to produce a noisy representation $\mathbf{z} = [\mathbf{z}^{(\mathbf{x})}, \mathbf{z}^{(\mathbf{h})}]$ and then learns a generative denoising process using the newly-proposed GCPNet++ model (see the Supplementary Methods of Appendix C.1.1), which desirably contains two distinct feature channels for scalar and vector features, respectively, and supports geometry-complete and chirality-aware message-passing [146].

As an extension of the DDPM framework [147] outlined in the Supplementary Methods of Appendix C.1.2, GCDM is designed to generate molecules in 3D while maintaining SE(3) equivariance, in contrast to previous methods that generate molecules solely in 1D [148], 2D [149], or 3D modalities without considering chirality [114, 103]. GCDM generates molecules by directly placing atoms in continuous 3D space and

assigning them discrete types, which is accomplished by modeling forward and reverse diffusion processes, respectively:

$$\underbrace{q(\mathbf{z}_{1:T}|\mathbf{z}_0)}_{\text{Forward}} = \prod_{t=1}^{T} q(\mathbf{z}_t|\mathbf{z}_{t-1}) \qquad \underbrace{p_{\mathbf{\Phi}}(\mathbf{z}_{0:T-1}|\mathbf{z}_T)}_{\text{Reverse}} = \prod_{t=1}^{T} p_{\mathbf{\Phi}}(\mathbf{z}_{t-1}|\mathbf{z}_t)$$

Overall, these processes describe a latent variable model $p_{\mathbf{\Phi}}(\mathbf{z}_0) = \int p_{\mathbf{\Phi}}(\mathbf{z}_{0:T}) d\mathbf{z}_{1:T}$ given a sequence of latent variables $\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_T$ matching the dimensionality of the data $\mathcal{M} \sim p(\mathbf{z}_0)$. As illustrated in Figure 4.1, the forward process (directed from right to left) iteratively adds noise to an input, and the learned reverse process (directed from left to right) iteratively denoises a noisy input to generate new examples from the original data distribution. We will now proceed to formulate GCDM's joint diffusion process and its remaining practical details.

### 4.5.3 Joint molecular diffusion

Recall that our model's molecular graph inputs, $\mathcal{G}$, associate with each node a 3D position $\mathbf{x}_i \in \mathbb{R}^3$ and a feature vector $\mathbf{h}_i \in \mathbb{R}^h$. By way of adding random noise to these model inputs at each time step $t$ via a fixed, Markov chain variance schedule $\sigma_1^2, \sigma_2^2, \ldots, \sigma_T^2$, we can define a joint molecular diffusion process for equivariant atom coordinates $\mathbf{x}$ and invariant atom types $\mathbf{h}$ as the product of two distributions [114]:

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}_{xh}(\mathbf{z}_t|\alpha_t \mathbf{z}_{t-1}, \sigma_t^2 \mathbf{I}). \tag{4.1}$$

where $\mathcal{N}_{xh}$ serves as concise notation to denote the product of two normal distributions; the first distribution, $\mathcal{N}_x$, represents the noised node coordinates; the second distribution, $\mathcal{N}_h$, represents the noised node features; and $\alpha_t = \sqrt{1 - \sigma_t^2}$ following the variance preserving process of [147]. With $\alpha_{t|s} = \alpha_t / \alpha_s$ and $\sigma_{t|s}^2 = \sigma_t^2 - \alpha_{t|s} \sigma_s^2$ for any $t > s$, we can directly obtain the noisy data distribution $q(\mathbf{z}_t|\mathbf{z}_0)$ at any time step

$t$:

$$q(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}_{xh}(\mathbf{z}_t|\alpha_{t|0}\mathbf{z}_0, \sigma^2_{t|0}\mathbf{I}). \tag{4.2}$$

Bayes Theorem then tells us that if we then define $\boldsymbol{\mu}_{t\to s}(\mathbf{z}_t, \mathbf{z}_0)$ and $\sigma_{t\to s}$ as

$$\boldsymbol{\mu}_{t\to s}(\mathbf{z}_t, \mathbf{z}_0) = \frac{\alpha_s \sigma^2_{t|s}}{\sigma^2_t}\mathbf{z}_0 \ + \frac{\alpha_{t|s}\sigma^2_s}{\sigma^2_t}\mathbf{z}_t \text{ and } \ \sigma_{t\to s} = \frac{\sigma_{t|s}\sigma_s}{\sigma_t},$$

we have that the inverse of the noising process, the true denoising process, is given by the posterior of the transitions conditioned on $\mathcal{M} \sim \mathbf{z}_0$, a process that is also Gaussian [114]:

$$q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{z}_0) = \mathcal{N}(\mathbf{z}_s|\boldsymbol{\mu}_{t\to s}(\mathbf{z}_t, \mathbf{z}_0), \sigma^2_{t\to s}\mathbf{I}). \tag{4.3}$$

### 4.5.4 Parametrization of the reverse process

**Noise parametrization.** We now need to define the learned generative reverse process that denoises pure noise into realistic examples from the original data distribution. Towards this end, we can directly use the noise posteriors $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{z}_0)$ of Eq. C.12 within the Supplementary Methods of Appendix C.1.2 after sampling $\mathbf{z}_0 \sim (\mathcal{M} = [\mathbf{x}, \mathbf{h}])$. However, to do so, we must replace the input variables $\mathbf{x}$ and $\mathbf{h}$ with the approximations $\hat{\mathbf{x}}$ and $\hat{\mathbf{h}}$ predicted by the denoising neural network $\boldsymbol{\Phi}$:

$$p_{\boldsymbol{\Phi}}(\mathbf{z}_s|\mathbf{z}_t) = \mathcal{N}_{xh}(\mathbf{z}_s|\boldsymbol{\mu}_{\boldsymbol{\Phi}_{t\to s}}(\mathbf{z}_t, \tilde{\mathbf{z}}_0), \sigma^2_{t\to s}\mathbf{I}), \tag{4.4}$$

where the values for $\tilde{\mathbf{z}}_0 = [\hat{\mathbf{x}}, \hat{\mathbf{h}}]$ depend on $\mathbf{z}_t$, $t$, and the denoising neural network $\boldsymbol{\Phi}$. GCDM then parametrizes $\boldsymbol{\mu}_{\boldsymbol{\Phi}_{t\to s}}(\mathbf{z}_t, \tilde{\mathbf{z}}_0)$ to predict the noise $\hat{\boldsymbol{\epsilon}} = [\hat{\boldsymbol{\epsilon}}^{(x)}, \hat{\boldsymbol{\epsilon}}^{(h)}]$, which represents the noise individually added to $\hat{\mathbf{x}}$ and $\hat{\mathbf{h}}$. We can then use the predicted $\hat{\boldsymbol{\epsilon}}$ to derive:

$$\tilde{\mathbf{z}}_0 = [\hat{\mathbf{x}}, \hat{\mathbf{h}}] = \mathbf{z}_t/\alpha_t - \hat{\boldsymbol{\epsilon}}_t \cdot \sigma_t/\alpha_t. \tag{4.5}$$

**Invariant likelihood.** Ideally, we desire for a 3D molecular diffusion model to assign the same likelihood to a generated molecule even after arbitrarily rotating or translating it in 3D space. To ensure the model achieves this desirable property for $p_{\mathbf{\Phi}}(\mathbf{z}_0)$, we can leverage the insight that an invariant distribution composed of an equivariant transition function yields an invariant distribution [116, 103, 114]. Moreover, to address the translation invariance issue raised by [116] in the context of handling a distribution over 3D coordinates, we adopt the zero center of gravity trick proposed by [103] to define $\mathcal{N}_x$ as a normal distribution on the subspace defined by $\sum_i \mathbf{x}_i = \mathbf{0}$. In contrast, to handle node features $\mathbf{h}_i$ that are invariant to roto-translations, we can instead use a conventional normal distribution $\mathcal{N}$. As such, if we parametrize the transition function $p_{\mathbf{\Phi}}$ using an SE(3)-equivariant neural network after using the zero center of gravity trick of [103], the model will have achieved the desired likelihood invariance property.

### 4.5.5 Geometry-complete denoising network

Crucially, to satisfy the desired likelihood invariance property described in Section 4.5.4 while optimizing for model expressivity and runtime, GCDM parametrizes the denoising neural network $\mathbf{\Phi}$ using GCPNet++, an enhanced version of the SE(3)-equivariant GCPNet algorithm [146], that we propose in the Supplementary Methods of Appendix C.1.1. Notably, GCPNet++ learns both scalar (invariant) and vector (equivariant) node and edge features through a chirality-sensitive graph message passing procedure, which enables GCDM to denoise its noisy molecular graph inputs using not only noisy scalar features but also noisy vector features that are derived directly from the noisy node coordinates $\mathbf{z}^{(\mathbf{x})}$ (i.e., $\psi(\mathbf{z}^{(\mathbf{x})})$). We empirically find that incorporating such noisy vectors considerably increases GCDM's representation capacity for 3D graph denoising.

### 4.5.6 Optimization objective

Following previous works on diffusion models [147, 114, 121], the noise parametrization chosen for GCDM yields the following model training objective:

$$\mathcal{L}_t = \mathbb{E}_{\epsilon_t \sim \mathcal{N}_{xh}(0,1)} \left[ \frac{1}{2} w(t) \|\epsilon_t - \hat{\epsilon}_t\|^2 \right], \tag{4.6}$$

where $\hat{\epsilon}_t$ is the denoising network's noise prediction for atom types and coordinates as described above and where we empirically choose to set $w(t) = 1$ for the best possible generation results. Additionally, GCDM permits a negative log-likelihood computation using the same optimization terms as [114], for which we refer interested readers to the Supplementary Methods of Appendices C.1.2, C.1.2, and C.1.2.

# Chapter 5

# GEOMETRIC FLOW MATCHING FOR GENERATIVE PROTEIN-LIGAND DOCKING AND AFFINITY PREDICTION

Adapted from Alex Morehead and Jianlin Cheng. "FlowDock: Geometric Flow Matching for Generative Protein-Ligand Docking and Affinity Prediction". *Intelligent Systems for Molecular Biology & Bioinformatics* (ISMB 2025).

## 5.1 ABSTRACT

Powerful generative AI models of protein-ligand structure have recently been proposed, but few of these methods support both flexible protein-ligand docking and affinity estimation. Of those that do, none can directly model multiple binding ligands concurrently or have been rigorously benchmarked on pharmacologically relevant drug targets, hindering their widespread adoption in drug discovery efforts. In this chapter, we propose FlowDock, the first deep geometric generative model based on conditional flow matching that learns to directly map unbound (apo) structures to their bound (holo) counterparts for an arbitrary number of binding ligands. Furthermore, FlowDock provides predicted structural confidence scores and binding affinity values with each of its generated protein-ligand complex structures, enabling fast virtual screening of new (multi-ligand) drug targets. For the well-known PoseBusters Benchmark dataset, FlowDock outperforms single-sequence AlphaFold 3 with a 51% blind docking success rate using unbound (apo) protein

input structures and without any information derived from multiple sequence alignments, and for the challenging new DockGen-E dataset, FlowDock outperforms single-sequence AlphaFold 3 and matches single-sequence Chai-1 for binding pocket generalization. Additionally, in the ligand category of the 16th community-wide Critical Assessment of Techniques for Structure Prediction (CASP16), FlowDock ranked among the top-5 methods for pharmacological binding affinity estimation across 140 protein-ligand complexes, demonstrating the efficacy of its learned representations in virtual screening. Source code, data, and pre-trained models are available at `https://github.com/BioinfoMachineLearning/FlowDock`.

## 5.2 INTRODUCTION

Interactions between proteins and small molecules (ligands) drive many of life's fundamental processes and, as such, are of great interest to biochemists, biologists, and drug discoverers. Historically, elucidating the structure, and therefore the function, of such interactions has required that considerable intellectual and financial resources be dedicated to determining the interactions of a single biomolecular complex. For example, techniques such as X-ray diffraction and cryo-electron microscopy have traditionally been effective in biomolecular structure determination, however, resolving even a single biomolecule's crystal structure can be extremely time and resource-intensive. Recently, new machine learning (ML) methods such as AlphaFold 3 (AF3) [27] have been proposed for directly predicting the structure of an arbitrary biomolecule from its primary sequence, offering the potential to expand our understanding of life's molecules and their implications in disease, energy research, and beyond.

Although powerful models of general biomolecular structure are compelling, they currently do not provide one with an estimate of the binding affinity of a predicted protein-ligand complex, which may indicate whether a pair of molecules truly bind to each other *in vivo*. It is desirable to predict both the structure of a protein-

ligand complex and the binding affinity between them via one single ML system [150]. Moreover, recent generative models of biomolecular structure are primarily based on noise schedules following Gaussian diffusion model methodology which, albeit a powerful modeling framework, lacks interpretability in the context of biological studies of molecular interactions. In this work, we aim to address these concerns with a new *state-of-the-art* hybrid (structure & affinity prediction) generative model called FlowDock for flow matching-based protein-ligand structure prediction and binding affinity estimation, which allows one to interpretably inspect the model's structure prediction trajectories to interrogate its common molecular interactions and to screen drug candidates quickly using its predicted binding affinities.

### 5.2.1 Related work

**Molecular docking with deep learning.** Over the last few years, deep learning (DL) algorithms (in particular geometric variants) have emerged as a popular methodology for performing end-to-end differentiable molecular docking. Models such as EquiBind [108] and TankBind [151] initiated a wave of interest in researching graph-based approaches to modeling protein-ligand interactions, leading to many follow-up works. Important to note is that most of such DL-based docking models were designed to supplement conventional modeling methods for protein-ligand docking such as AutoDock Vina [152] which are traditionally slow and computationally expensive to run for many protein-ligand complexes yet can achieve high accuracy with crystal input structures and ground-truth binding pocket annotations.

**Generative biomolecular modeling.** The potential of generative modeling in capturing intricate molecular details in structural biology such as protein-ligand interactions during molecular docking [99] has recently become a research focus of ambitious biomolecular modeling efforts such as AF3 [27], with several open-source spin-offs of this algorithm emerging [153, 154].

**Flow matching.** In the machine learning community, generative modeling with flow matching [155, 156] has recently become an appealing generalization of diffusion generative models [147, 157], enabling one to transport samples between arbitrary distributions for compelling applications in computer vision [158], computational biology [159], and beyond. As a closely related concurrent work (as our method was developed for the CASP16 competition starting in May 2024 [160]), [161] recently introduced and evaluated an unbalanced flow matching procedure for pocket-based flexible docking. However, the authors' proposed approach mixes diffusion and flow matching noise schedules with geometric product spaces in an unintuitive manner, and neither source code nor data for this work are publicly available for benchmarking comparisons. In Section 5.3.3, we describe flow matching in detail.

**Contributions.** In light of such prior works, our contributions in this manuscript are as follows:

- We introduce the *first* simple yet state-of-the-art *hybrid* generative flow matching model capable of quickly and accurately predicting protein-ligand complex structures *and* their binding affinities, with source code and model weights freely available.
- We rigorously validate our proposed methodology using standardized benchmarking data for protein-ligand complexes, with our method ranking as a more accurate and generalizable structure predictor than (single-sequence) AF3.
- Our method ranked as a top-5 binding affinity predictor for the 140 pharmaceutically relevant drug targets available in the 2024 community-wide CASP16 ligand prediction competition.
- We release one of the largest ML-ready datasets of apo-to-holo protein structure mappings based on high-accuracy predicted protein structures, which enables training new models on comprehensive biological data for distributional

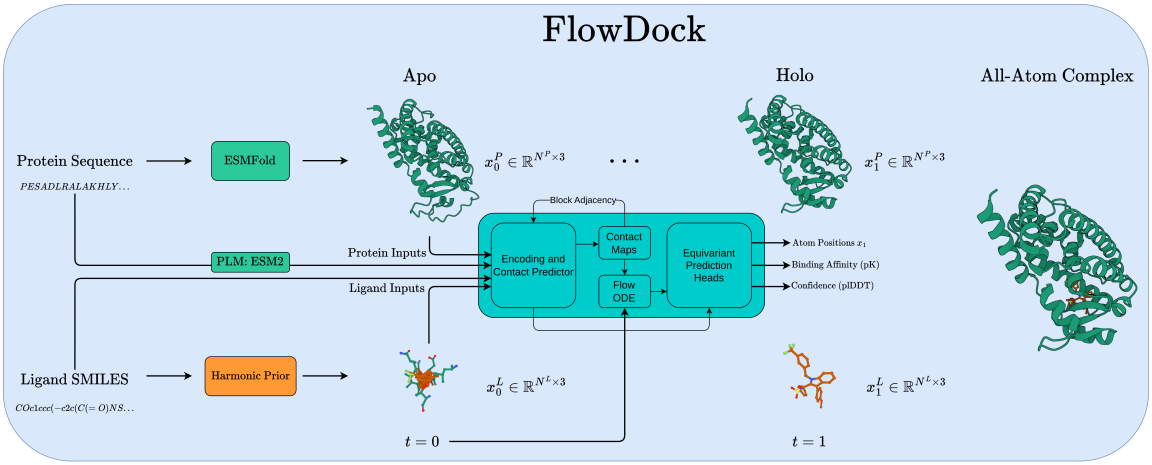



Figure 5.1: An overview of biomolecular distribution modeling with FlowDock.

biomolecular structure modeling.

## 5.3 METHODS

The goal of this work is to jointly predict protein-ligand complex structures and their binding affinities with minimal computational overhead to facilitate drug discovery. In Sections 5.3.1 and 5.3.2, we briefly outline how FlowDock achieves this and how its key notation is defined. We then describe FlowDock's training and sampling procedures in Sections 5.3.3-5.3.6.

### 5.3.1 Overview

Figure 5.1 illustrates how FlowDock uses geometric flow matching to predict flexible protein-ligand structures and binding affinities. At a high level, FlowDock accepts both (multi-chain) protein sequences and (multi-fragment) ligand SMILES strings as its primary inputs, which it uses to predict an unbound (apo) state of the protein sequences using ESMFold [25] and to sample from a harmonic ligand prior distribution [162] to initialize the ligand structures using biophysical constraints based on their specified bond graphs. Notably, users can also specify the initial protein structure using one produced by another bespoke method (e.g., AF3 which we use in

certain experiments). With these initial structures representing the complex's state at time $t = 0$, FlowDock employs conditional flow matching to produce fast structure generation trajectories. After running a small number of integration timesteps (e.g., 40 in our experiments), the complex's state arrives at time $t = 1$, i.e., the model's estimate of the bound (holo) protein-ligand heavy-atom structure. At this point, FlowDock runs confidence and binding affinity heads to predict structural confidence scores (i.e., plDDT) and binding affinities of the predicted complex structure, to rank-order the model's generated samples.

### 5.3.2 Notation

Let $\boldsymbol{x}_0$ denote the unbound (apo) state of a protein-ligand complex structure, representing the heavy atoms of the protein and ligand structures as $\boldsymbol{x}_0^P \in \mathbb{R}^{N^P \times 3}$ and $\boldsymbol{x}_0^L \in \mathbb{R}^{N^L \times 3}$, respectively, where $N^P$ and $N^L$ are the numbers of protein and ligand heavy atoms. Similarly, we denote the corresponding bound (holo) state of the complex as $\boldsymbol{x}_1$. Further, let $\boldsymbol{s}^P \in \{1, \ldots, 20\}^{S^P}$ denote the type of each amino acid residue in the protein structure, where $S^P$ represents the protein's sequence length. To generate bound (holo) structures, we define a flow model $v_\theta$ that integrates the ordinary differential equation (ODE) it defines from time $t = 0$ to $t = 1$.

### 5.3.3 Riemannian manifolds and conditional flow matching

In manifold theory, an $n$-dimensional manifold $\mathcal{M}$ represents a topological space equivalent to $\mathbb{R}^n$. In the context of Riemannian manifold theory, each point $\boldsymbol{x} \in \mathcal{M}$ on a Riemannian manifold is associated with a tangent space $\mathcal{T}_{\boldsymbol{x}}$. Conveniently, a Riemannian manifold is equipped with a metric $g_{\boldsymbol{x}} : \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \to \mathbb{R}$ that permits the definition of geometric quantities on the manifold such as distances and geodesics (i.e., shortest paths between two points on the manifold). Subsequently, Riemannian manifolds allow one to define on them probability densities $\int_{\mathcal{M}} \rho(\boldsymbol{x}) d\boldsymbol{x} = 1$ where $\rho :$

$\mathcal{M} \to \mathbb{R}_+$ are continuous, non-negative functions. Such probability densities give rise to interpolative probability paths $\rho_t : [0,1] \to \mathbb{P}(\mathcal{M})$ between probability distributions $\rho_0, \rho_1 \in \mathbb{P}(\mathcal{M})$, where $\mathbb{P}(\mathcal{M})$ is defined as the space of probability distributions on $\mathcal{M}$ and the interpolation in probability space between distributions is indexed by the continuous parameter $t$.

Here, we refer to $\psi_t : \mathcal{M} \to \mathcal{M}$ as a *flow* on $\mathcal{M}$. Such a flow serves as a solution to the ODE: $\frac{d}{dt}\psi_t(\boldsymbol{x}) = u_t(\psi_t(\boldsymbol{x}))$ [163] which allows one to *push forward* the probability trajectory $\rho_0 \to \rho_1$ to $\rho_t$ using $\psi_t$ as $\rho_t = [\psi_t]_{\#}(\rho_0)$, with $\psi_0(\boldsymbol{x}) = \boldsymbol{x}$ for $u : [0,1] \times \mathcal{M} \to \mathcal{M}$ (i.e., a smooth time-dependent vector field [164]). This insight allows one to perform *flow matching* (FM) [155] between $\rho_0$ and $\rho_1$ by learning a continuous normalizing flow [165] to approximate the vector field $u_t$ with the parametric $v_\theta$. With $\rho_0 = \rho_{prior}$ and $\rho_1 = \rho_{data}$, we have that $\rho_t$ advantageously permits *simulation-free* training. Although it is not possible to derive a closed form for $u_t$ (which generates $\rho_t$) with the traditional flow matching (FM) training objective, a *conditional* flow matching (CFM) training objective remains tractable by marginalizing conditional vector fields as $u_t(\boldsymbol{x}) := \int_{\mathcal{M}} u_t(\boldsymbol{x}|\boldsymbol{z}) \frac{\rho_t(\boldsymbol{x}_t|\boldsymbol{z})q(\boldsymbol{z})}{\rho_t(\boldsymbol{x})} d\boldsymbol{z}$, where $q(\boldsymbol{z})$ represents one's chosen coupling distribution (by default the independent coupling $q(\boldsymbol{z}) = q(\boldsymbol{x}_0)q(\boldsymbol{x}_1)$) between $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ via the conditioning variable $\boldsymbol{z}$. For Riemannian CFM (RCFM) [155], the corresponding training objective, with $t \sim \mathcal{U}(0,1)$, is:

$$\mathcal{L}_{RCFM}(\theta) = \mathbb{E}_{t, q(\boldsymbol{z}), \rho_t(\boldsymbol{x}_t|\boldsymbol{z})} \| v_\theta(\boldsymbol{x}_t, t) - u_t(\boldsymbol{x}_t|\boldsymbol{z}) \|_g^2, \tag{5.1}$$

where [156] have fortuitously shown that the gradients of FM and CFM are identical. As such, to transport samples of the prior distribution $\rho_0$ to the target (data) distribution $\rho_1$, one can sample from $\rho_0$ and use $v_\theta$ to run the corresponding ODE forward in time. In the remainder of this work, we will focus specifically on the 3-manifold $\mathbb{R}^3$.

### 5.3.4 Prior distributions

With flow matching defined, in this section, we describe how we use a bespoke mixture of prior distributions ($\rho_0^P$ and $\rho_0^L$) to sample initial (unbound) protein and ligand structures for binding (holo) structure generation targeting our data distribution of crystal protein-ligand complex structures $\rho_1$. In Section 5.4.1, we ablate this mixture to understand its empirical strengths.

**ESMFold protein prior.** To our best knowledge, FlowDock is among the *first* methods-concurrently with [161]-to explore using structure prediction models with flow matching to represent the unbound state of an arbitrary protein sequence. In contrast to [161], we formally define a *distribution* of unbound (apo) protein structures using the single-sequence ESMFold model as $\rho_0^P(\boldsymbol{x}_0^P) \propto \text{ESMFold}(\boldsymbol{s}^P) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma)$, which encourages our model to learn more than a strict mapping between protein apo and holo point masses. Based on previous works developing protein generative models [166], during training we apply $\epsilon \sim \mathcal{N}(0, \sigma = 1e^{-4})$ to both $\boldsymbol{x}_0^P$ and $\boldsymbol{x}_1^P$ to discourage our model from overfitting to computational or experimental noise in its training data. It is important to note that this additive noise for protein structures is not a general substitute for generating a full conformational ensemble of each protein, but to avoid the excessively high computational resource requirements of running protein dynamics methods such as AlphaFlow [162] for each protein, we empirically find noised ESMFold structures to be a suitable surrogate.

**Harmonic ligand prior.** Inspired by the FlowSite model for multi-ligand binding site design [167], FlowDock samples initial ligand conformations using a harmonic prior distribution constrained by the bond graph defined by one's specified ligand SMILES strings. This prior can be sampled as a modified Gaussian distribution via $\rho_0^L(\boldsymbol{x}_0^L) \propto exp(-\frac{1}{2}\boldsymbol{x}_0^{L^T}\boldsymbol{L}\boldsymbol{x}_0^L)$ where $\boldsymbol{L}$ denotes a ligand bond graph's Laplacian matrix defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$, with $\boldsymbol{A}$ being the graph's adjacency matrix and $\boldsymbol{D}$ being its degree matrix. Similarly to our ESMFold protein prior, we subsequently

apply $\epsilon \sim \mathcal{N}(0, \sigma = 1e^{-4})$ to $\boldsymbol{x}_1^L$ during training.

### 5.3.5 Training

We describe FlowDock's structure parametrization, optimization procedure, and the curation and composition of its new training dataset in the following sections. Further, we provide training and inference pseudocode in the Supplementary Materials of Appendix D.1.

**Parametrizing protein-ligand complexes with geometric flows.** Based on our experimental observations of the difficulty of scaling up intrinsic generative models [168] that operate on geometric product spaces, FlowDock instead parametrizes 3D protein-ligand complex structures as attributed geometric graphs [107] representing the heavy atoms of each complex's protein and ligand structures. The main benefit of a heavy atom parametrization is that it can considerably simplify the optimization of a flow model $v_\theta$ by allowing one to define its primary loss function as simply as a CondOT path [169, 162]:

$$\mathcal{L}_{\mathbb{R}^3}(\theta) = \mathbb{E}_{t, q(\boldsymbol{z}), \rho_t(\boldsymbol{x}_t|\boldsymbol{z})} \| v_\theta(\boldsymbol{x}_t, t) - \boldsymbol{x}_1) \|^2, \tag{5.2}$$

with the conditional probability path $\rho_t$ chosen as

$$\rho_t(\boldsymbol{x}|\boldsymbol{z}) = \rho_t(\boldsymbol{x}|\boldsymbol{x}_0, \boldsymbol{x}_1) = (1-t) \cdot \boldsymbol{x}_0 + t \cdot \boldsymbol{x}_1, \quad \boldsymbol{x}_0 \sim \rho_0(\boldsymbol{x}_0) \tag{5.3}$$

The challenge introduced by this atomic parametrization is that it necessitates the development of an efficient neural architecture that can scalably process all-atom input structures without the exhaustive computational overhead of generative models such as AF3. Fortunately, one such architecture satisfies this requirement, namely, one recently introduced by [29] with the NeuralPLexer model which encodes protein language model (PLM) sequence embeddings and ligand SMILES strings to itera-

tively decode block diagonal contact maps to condition a flow ODE for equivariant coordinates and auxiliary predictions. As such, inspired by how the AlphaFlow model was fine-tuned from the base AlphaFold 2 (AF2) architecture using flow matching, to train FlowDock we explored fine-tuning the NeuralPLexer architecture to represent our vector field estimate $v_\theta$ as illustrated in Figure 5.1. Uniquely, we empirically found this idea to work best by fine-tuning the architecture's score head, which was originally trained with a denoising score matching objective for *diffusion*-based structure sampling, instead using Eqs. 5.2 and 5.3. Moreover, we fine-tune all of NeuralPLexer's remaining intermediate weights and prediction heads including a dedicated confidence head redesigned to predict binding affinities, with the exception of its original confidence head which remains frozen at all points during training.

**PDBBind-E Data Curation.** To train FlowDock with resolved protein-ligand structures and binding affinities, we prepared PDBBind-E, an enhanced version of the PDBBind 2020-based training dataset proposed by [170] for training recent DL docking methods such as DiffDock-L. To curate PDBBind-E, we collected 17,743 crystal complex structures contained in the PDBBind 2020 dataset and 47,183 structures of the Binding MOAD [133] dataset splits introduced by [170] (n.b., which maintain the validity of our benchmarking results in Section 5.4 according to time and ligand-based similarity cutoffs) and predicted the structure of these (multi-chain) protein sequences in each dataset split using ESMFold. To optimally align each predicted protein structure with its corresponding crystal structure, we performed a weighted structural alignment optimizing for the distances of the predicted protein residues' C$\alpha$ atoms to the crystal heavy atom positions of the complex's binding ligand, similar to [170]. After dropping complexes for which the crystal structure contained protein sequence gaps caused by unresolved residues, the total number of PDBBind and Binding MOAD predicted complex structures remaining was 17,743 and 46,567, respectively.

**Generalized unbalanced flow matching.** We empirically observed the challenges of naively training flexible docking models like FLOWDOCK without any adjustments to the sampling of their training data. Accordingly, we concurrently developed a generalized version of *unbalanced* flow matching [161] by defining our coupling distribution $q(\boldsymbol{z})$ as

$$q(\boldsymbol{x}_0, \boldsymbol{x}_1) \propto q_0(\boldsymbol{x}_0) q_1(\boldsymbol{x}_1) \mathbb{I}_{c(\boldsymbol{x}_0, \boldsymbol{x}_1) \in c_{\mathbb{A}}}, \tag{5.4}$$

where $c_{\mathbb{A}}$ is defined as a set of apo-to-holo assessment filters measuring the structural similarity of the unbound (apo) and bound (holo) protein structures (n.b., not simply their binding pockets) in terms of their root mean square deviation (RMSD) and TM-score [171] following optimal structural alignment (as used in constructing PDBBind-E). Effectively, we sample independent examples from $q_0$ and $q_1$ and reject these paired examples if $c(\boldsymbol{x}_0, \boldsymbol{x}_1) < c_{\mathbb{A}_{TM}}$ or $c(\boldsymbol{x}_0, \boldsymbol{x}_1) \geq c_{\mathbb{A}_{RMSD}}$ (n.b., we use $c_{\mathbb{A}_{TM}} = 0.7$ and $c_{\mathbb{A}_{RMSD}} = 5$Å as well as other length-based criteria in our experiments, please see our code for full details).

### 5.3.6 Sampling

By default, we apply $i = 40$ timesteps of an Euler solver to integrate FLOWDOCK's learned ODE $v_\theta$ forward in time for binding (holo) structure generation. Specifically, to generate structures, we propose to integrate a Variance Diminishing ODE (VD-ODE) that uses $v_\theta$ as

$$\boldsymbol{x}_{n+1} = clamp(\frac{1-s}{1-t} \cdot \eta) \cdot \boldsymbol{x}_n + clamp((1 - \frac{1-s}{1-t}) \cdot \eta) \cdot v_\theta(\boldsymbol{x}_n, t), \tag{5.5}$$

where $n$ represents the current integer timestep, allowing us to define $t = \frac{n}{i}$ and $s = \frac{n+1}{i}$; $\eta = 1.0$ in our experiments; and *clamp* ensures both the LHS and RHS of Eq. 5.5 are lower and upper bounded by $1e^{-6}$ and $1 - 1e^{-6}$, respectively. We experimented with different values of $\eta$ yet ultimately settled on 1.0 since this yielded

FlowDock's best performance for structure and affinity prediction. Intuitively, this VD-ODE solver limits the high levels of variance present in the model's predictions $v_\theta$ during early timesteps by sharply interpolating towards $v_\theta$ in later timesteps.

## 5.4 RESULTS

### 5.4.1 PoseBench protein-ligand docking

**PoseBusters Benchmark set.** In Figures 5.2 and 5.3, we illustrate the performance of each baseline method for protein-ligand docking and protein conformational modification with the commonly-used PoseBusters Benchmark set [112], provided by version 0.6.0 of the PoseBench protein-ligand benchmarking suite [172], which consists of 308 distinct protein-ligand complexes released after 2020. It is important to note that this benchmarking set can be considered a moderately difficult challenge for methods trained on recent collections of data derived from the Protein Data Bank (PDB) [173] such as PDBBind 2020 [174], as all of these 308 protein-ligand complexes are not contained in the most common training splits of such PDB-based data collections [112] (with the exception of AF3 which uses a cutoff date of September 30, 2021). Moreover, as described by [112], a subset of these complexes also have very low protein sequence similarity to such training splits.

Figure 5.2 shows that FlowDock consistently improves over the original NeuralPLexer model's docking success rate in terms of its structural and chemical accuracy (as measured by the RMSD $\leq 2$Å & PB-Valid metric [112]) and inter-run stability (as measured by the error bars listed). Notably, FlowDock achieves a 10% higher docking success rate than NeuralPLexer without any structural energy minimization driven by molecular dynamics software [175], and with energy minimization its docking success rate increases to 51%, outperforming single-sequence AF3 and achieving second-best performance on this dataset compared to single-sequence Chai-1 [153]. Important to note is that Chai-1, like AF3, is a 10x larger model trained for one month

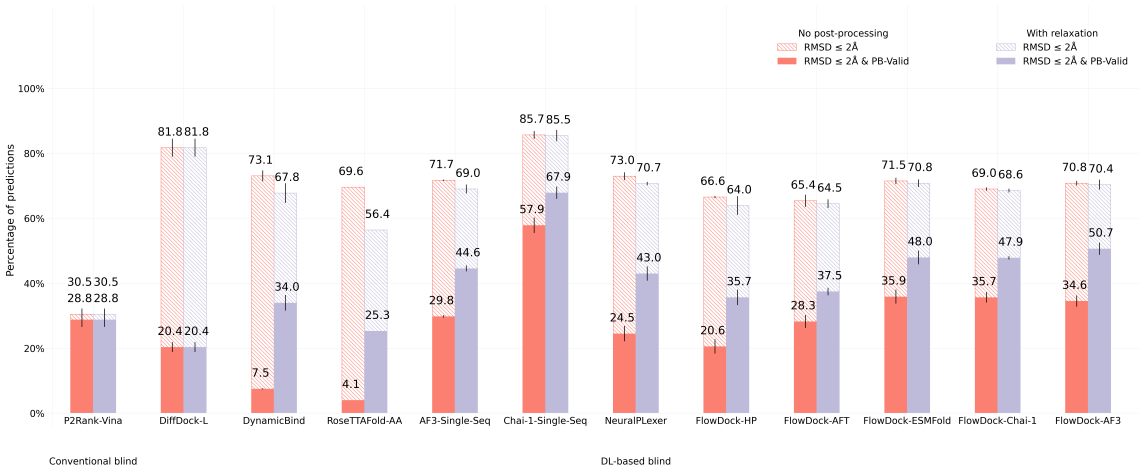
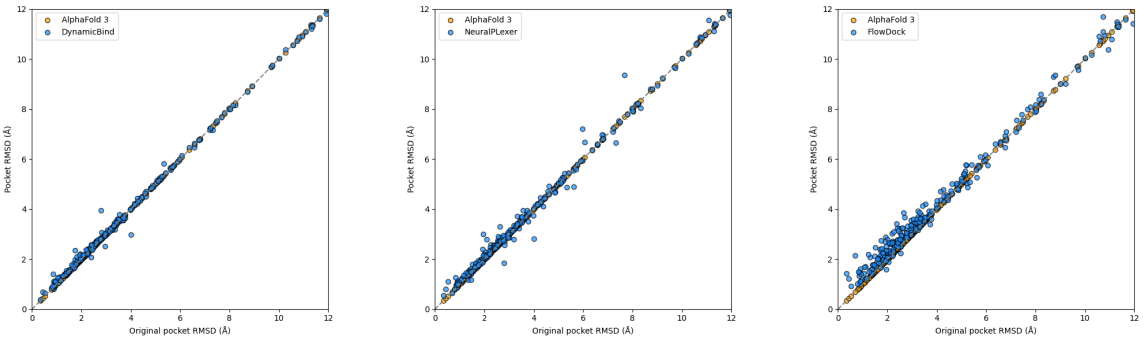



Figure 5.2: Protein-ligand docking success rates of each baseline method on the PoseBusters Benchmark set (n=308). Error bars: 3 runs.




**DynamicBind** (Binding Pocket Conformation RMSDs)

**NeuralPLexer** (Binding Pocket Conformation RMSDs)

**FlowDock** (Binding Pocket Conformation RMSDs)

Figure 5.3: Comparison of each flexible docking method's protein conformational changes made for the PoseBusters Benchmark set (n=308).

using 128 NVIDIA A100 80GB GPUs on more than twice as much data in the PDB deposited up to 2021, whereas FlowDock is trained using only 4 80GB H100 GPUs for one week, representing a 32x reduction in GPU hours required for training. Additionally, FlowDock outperforms the *hybrid* flexible docking method DynamicBind [28] by more than 16%, which is a comparable model in terms of its size, training, and downstream capabilities for drug discovery. Our results with ablated versions of FlowDock trained instead with a protein harmonic prior (FlowDock-HP) or with affinity prediction frozen until a fine-tuning phase (FlowDock-AFT) highlight

that the protein ESMFold prior the base FlowDock model employs has imbued it with meaningful structural representations for accurate ligand binding structure prediction that are robust to changes in the source method of FlowDock's predicted protein input structures (e.g., FlowDock-ESMFold vs. FlowDock-Chai-1 vs. FlowDock-AF3), providing users with multiple structure prediction options (e.g., ESMFold for faster and commercially available prediction inputs).

A surprising finding illustrated in Figure 5.3 is that no method can consistently improve the binding pocket RMSD of AF3's initial protein structural conformations, which contrasts with the results originally reported for flexible docking methods such as DynamicBind which used structures predicted by AF2 [15] in its experiments. From this figure, we observe that DynamicBind and NeuralPLexer both infrequently modify AF3's initial binding pocket structure, whereas FlowDock often modifies the pocket structure during ligand binding. The former two methods occasionally improve largely-correct initial pocket conformations by ∼1Å, whereas FlowDock primarily does so for mostly-incorrect initial pockets.

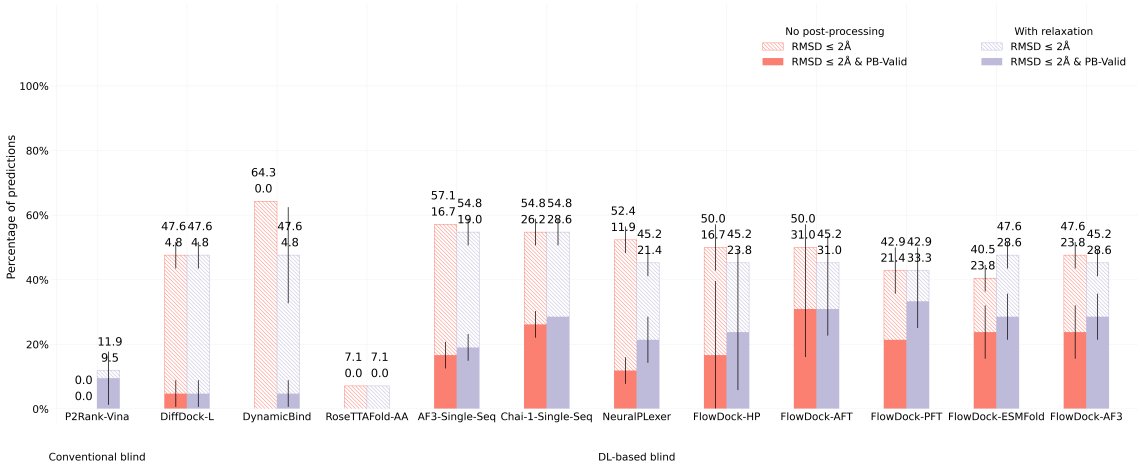
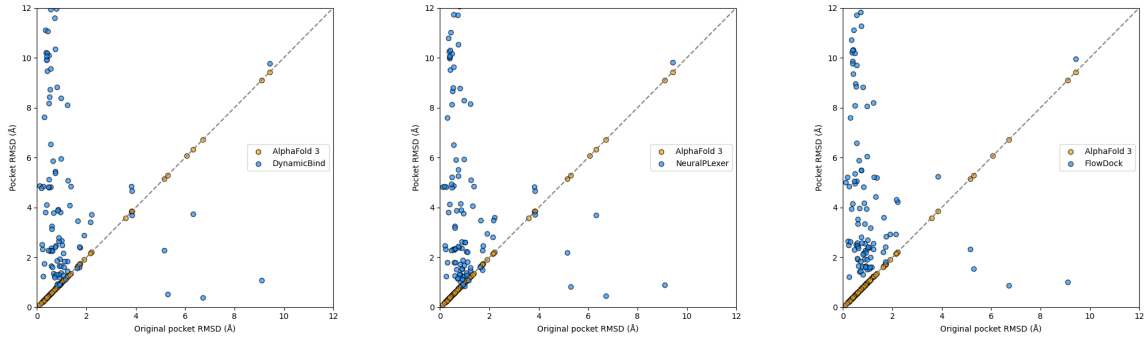



Figure 5.4: Protein-ligand docking success rates of each baseline method on the DockGen-E set (n=14). Error bars: 3 runs.

**DockGen-E set.** To assess the generalization capabilities of each baseline method, in Figures 5.4 and 5.5, we report each method's protein-ligand docking and protein




**DynamicBind** (Binding Pocket Conformation RMSDs)

**NeuralPLexer** (Binding Pocket Conformation RMSDs)

**FlowDock** (Binding Pocket Conformation RMSDs)

Figure 5.5: Comparison of each flexible docking method's protein conformational changes made for the DockGen-E set (n=122).

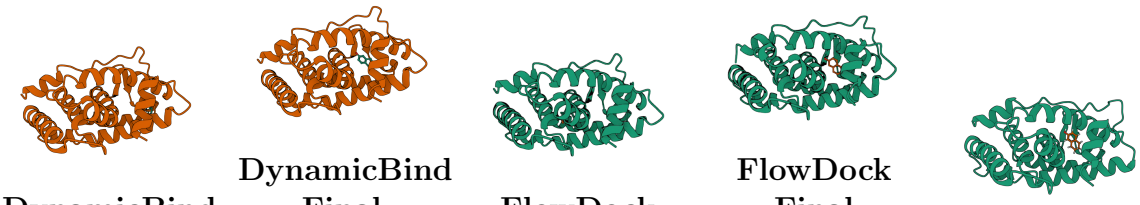
conformational modification performance for the novel (i.e., naturally rare) protein binding pockets found in the new DockGen-E dataset from PoseBench. Each of DockGen-E's protein-ligand complexes represents a distinct binding pocket that facilitates a unique biological function described by its associated ECOD domain identifier [170]. As our results for the DockGen-E dataset show in Figure 5.4, most DL-based docking or structure prediction methods have likely not been trained or overfitted to these binding pockets, as this dataset's best docking success rate achieved by any method is approximately 33%, much lower than the 68% best docking success rate achieved for the PoseBusters Benchmark set. We find further support for this phenomenon in Figure 5.5, where we see that all DL-based flexible docking methods find it challenging to avoid degrading the initial binding pocket state predicted by AF3 yet all methods can *restore* a handful of AF3 binding pockets to their bound (holo) form. This suggests that all DL methods (some more so than others) struggle to generalize to novel binding pockets, yet FlowDock achieves top performance in this regard by tying with single-sequence Chai-1. Further, to address this generalization issue, our preliminary results fine-tuning FlowDock for 48 hours using the new, diverse PLINDER [176] dataset (i.e., FlowDock-PFT), where we use the dataset's

Table 5.1: **Computational resource requirements.** The average structure prediction runtime (in seconds) and peak memory usage (in GB) of baseline methods on a 25% subset of the Astex Diverse dataset [177] using an NVIDIA 80GB A100 GPU for benchmarking (with baselines taken from [172]). The symbol - denotes a result that could not be estimated.

| Method | Runtime (s) | CPU Memory Usage (GB) | GPU Memory Usage (GB) |
|---|---|---|---|
| P2Rank-Vina | 1,283.70 | 9.62 | 0.00 |
| DiffDock-L | 88.33 | 8.99 | 70.42 |
| DynamicBind | 146.99 | 5.26 | 18.91 |
| RoseTTAFold-All-Atom | 3,443.63 | 55.75 | 72.79 |
| AF3 | 3,049.41 | - | - |
| AF3-Single-Seq | 58.72 | - | - |
| Chai-1-Single-Seq | 114.86 | 58.49 | 56.21 |
| NeuralPLexer | 29.10 | 11.19 | 31.00 |
| **FlowDock** | 39.34 | 11.98 | 25.61 |

crystal apo-to-holo mapped protein-ligand complex structures contained within its default PL50 training split and deposited in the PDB before 2018, suggest that comprehensively training new DL methods on diverse protein-ligand binding structures is a promising direction towards generalizable docking.

**Computational resources.** To formally measure the computational resources required to run each baseline method, in Table 5.1 we list the average runtime (in seconds) and peak CPU (GPU) memory usage (in GB) consumed by each method when running them on a 25% subset of the Astex Diverse dataset [177] (baseline results taken from [172]). Here, we notably find that FlowDock provides the second lowest computational runtime and GPU memory usage compared to all other DL methods, enabling one to use commodity computing hardware to quickly screen new drug candidates using combinations of FlowDock's predicted heavy-atom structures, confidence scores, and binding affinities.



**DynamicBind Init.** (ESMFold Structure) | **DynamicBind Final** (Affinity / plDDT: 5.72 / 0.83) | **FlowDock Init.** (ESMFold Structure) | **FlowDock Final** (Affinity / plDDT: 6.05 / 0.78) | **6I67 Crystal** (True Affinity: 6.70)

Figure 5.6: Comparison of DynamicBind and FlowDock's predicted structures (w/o hydrogens) and crystal PDBBind test example 6I67.

Table 5.2: **Binding affinity estimation using PDBBind test set.** For all methods, binding affinities were predicted in *one shot* using the commonly-used 363 PDBBind (ligand and time-split) test complexes (with splits and baselines from [28]). Results for FlowDock are reported as the mean and standard error of measurement ($n = 3$) of each metric over three independent runs. Note that, for historical reasons, the results for each version of FlowDock were obtained using ESMFold predicted protein input structures.

| Method | Pearson (↑) | Spearman (↑) | RMSE (↓) | MAE (↓) |
|---|---|---|---|---|
| GIGN | 0.286 | 0.318 | 1.736 | 1.330 |
| TransformerCPI | 0.470 | 0.480 | 1.643 | 1.317 |
| MONN | 0.545 | 0.535 | 1.371 | 1.103 |
| TankBind | 0.597 | 0.610 | 1.436 | 1.119 |
| DynamicBind (One-Shot) | 0.665 | 0.634 | 1.301 | 1.060 |
| FlowDock-HP | $0.577 \pm 0.001$ | $0.560 \pm 0.001$ | $1.516 \pm 0.001$ | $1.196 \pm 0.002$ |
| FlowDock-AFT | $0.663 \pm 0.003$ | $0.624 \pm 0.003$ | $1.392 \pm 0.005$ | $1.113 \pm 0.003$ |
| **FlowDock** | $\mathbf{0.705 \pm 0.001}$ | $\mathbf{0.674 \pm 0.002}$ | 1.363±0.003 | 1.067±0.003 |

### 5.4.2 PDBBind binding affinity estimation

In this section, we explore binding affinity estimation with FlowDock using the PDBBind 2020 test dataset (n=363) originally curated by [108], with benchmarking results shown in Table 5.2. Popular affinity prediction baselines listed in Table 5.2 such as TankBind [151] and DynamicBind [28] demonstrate that accurate affinity estimations are possible using hybrid DL models of protein-ligand structures and affinities. Here, we find that, as a hybrid deep generative model, FlowDock provides the best Pearson and Spearman's correlations compared to all other baselines includ-

ing FlowDock-HP (a fully harmonic variant of FlowDock) and FlowDock-AFT (an ESMFold prior variant trained first for structure prediction and then with affinity fine-tuning) and produces compelling root mean squared error (RMSE) and mean absolute error (MAE) rates compared to the previous state-of-the-art method DynamicBind. Referencing Table 5.1, we further note that FlowDock's average computational runtime per protein-ligand complex is more than 3 times lower than that of DynamicBind, demonstrating that FlowDock, to our best knowledge, is currently the *fastest* binding affinity estimation method to match or exceed DynamicBind's level of accuracy for predicting binding affinities using the PDBBind 2020 dataset.

In Figure 5.6, we provide an illustrative example of a protein-ligand complex in the PDBBind test set (6I67) for which FlowDock predicts notably more accurate complex structural motions and binding affinity values than the hybrid DL method DynamicBind, importantly recognizing that the right-most protein loop domain should be moved further to the right to facilitate ligand binding (see the Supplementary Materials of Appendix D.2 for an example of one of FlowDock's interpretable structure generation trajectories). One should note that, for historical reasons, our experiments with this PDBBind-based test set employed protein structures predicted by ESMFold (not AF3). In the next section, we explore an even more practical application of FlowDock's fast and accurate structure and binding affinity predictions in the CASP16 ligand prediction competition.

### 5.4.3 CASP16 protein-ligand binding affinity prediction

In Figure 5.7, we illustrate the performance of each predictor group for blind protein-ligand binding affinity prediction in the ligand category of the CASP16 competition held in summer 2024, in which pharmaceutically relevant binding ligands were the primary focus of this competition. Notably, FlowDock is the *only* hybrid (structure &

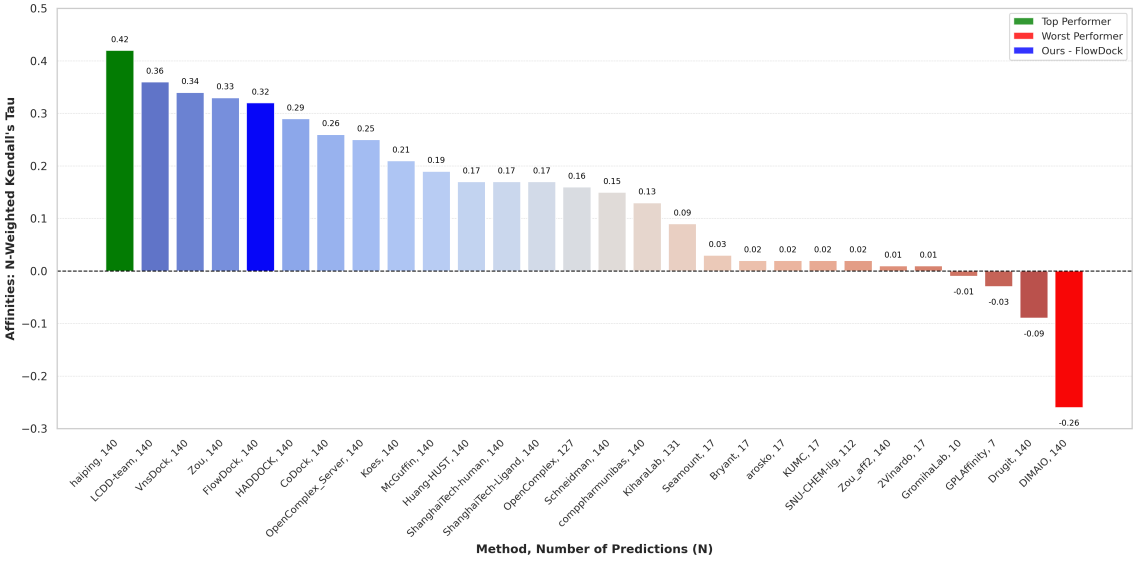



Figure 5.7: Protein-ligand binding affinity prediction rankings for the CASP16 ligand prediction category (n=140).

affinity prediction) ML method represented among the top-5 predictors, demonstrating the robustness of its knowledge of protein-ligand interactions. Namely, all other top prediction methods were trained specifically for binding affinity estimation assuming a predicted or crystal complex structure is provided. In contrast, in CASP16, we demonstrated the potential of using FlowDock to predict *both* protein-ligand structures and binding affinities and using its top-5 predicted structures' structural confidence scores to rank-order its top-5 binding affinity predictions (see the Supplementary Materials of Appendices D.3 and D.4 for FlowDock's e.g., CASP16 structure prediction results). Ranked 5th for binding affinity estimation, these results of the CASP16 competition demonstrate that this dual approach of predicting protein-ligand structures and binding affinities with a single DL model (FlowDock) yields compelling performance for virtual screening of pharmaceutically interesting molecular compounds.

## 5.5 DISCUSSION

In this chapter, we have presented FlowDock, a novel, state-of-the-art deep generative flow model for fast and accurate (hybrid) protein-ligand binding structure and affinity prediction. Benchmarking results suggest that FlowDock achieves structure prediction results better than single-sequence AF3 and comparable to single-sequence Chai-1 and outperforms existing hybrid models like DynamicBind across a range of binding ligands. Lastly, we have demonstrated the pharmaceutical virtual screening potential of FlowDock in the CASP16 ligand prediction competition, where it achieved top-5 performance. Future work could include retraining the model on larger and more diverse clusters of protein-ligand complexes, experimenting with new ODE solvers, or scaling up its parameter count to see if it displays any scaling law behavior for structure or affinity prediction. As a deep generative model for structural biology made available under an MIT license, we believe FlowDock takes a notable step forward towards fast, accurate, and broadly applicable modeling of protein-ligand interactions.

# Chapter 6

# DEEP LEARNING FOR PROTEIN-LIGAND DOCKING: ARE WE THERE YET?

Adapted from Alex Morehead, Nabin Giri, Jian Liu, Pawan Neupane, and Jianlin Cheng. "Deep Learning for Protein-Ligand Docking: Are We There Yet?". *AI for Science Workshop of the Forty-First International Conference on Machine Learning* (ICML 2024 AI4Science Spotlight).

## 6.1 ABSTRACT

The effects of ligand binding on protein structures and their *in vivo* functions carry numerous implications for modern biomedical research and biotechnology development efforts such as drug discovery. Although several deep learning (DL) methods and benchmarks designed for protein-ligand docking have recently been introduced, to date no prior works have systematically studied the behavior of the latest docking and structure prediction methods within the *broadly applicable* context of (1) using predicted (apo) protein structures for docking (e.g., for applicability to new proteins); (2) binding multiple (cofactor) ligands concurrently to a given target protein (e.g., for enzyme design); and (3) having no prior knowledge of binding pockets (e.g., for generalization to unknown pockets). To enable a deeper understanding of docking methods' real-world utility, in this chapter, we introduce PoseBench, the first comprehensive benchmark for *broadly applicable* protein-ligand

docking. PoseBench enables researchers to rigorously and systematically evaluate DL methods for apo-to-holo protein-ligand docking and protein-ligand structure prediction using *both* primary ligand and multi-ligand benchmark datasets, the latter of which we introduce for the first time to the DL community. Empirically, using PoseBench, we find that (1) DL co-folding methods generally outperform comparable conventional and DL docking baselines, yet popular methods such as AlphaFold 3 are still challenged by prediction targets with novel protein sequences; (2) certain DL co-folding methods are highly sensitive to their input multiple sequence alignments, while others are not; and (3) DL methods struggle to strike a balance between structural accuracy and chemical specificity when predicting novel or multi-ligand protein targets. Code, data, tutorials, and benchmark results are available at `https://github.com/BioinfoMachineLearning/PoseBench`.

## 6.2 INTRODUCTION

The field of drug discovery has long been challenged with a critical task: determining the structure of ligand molecules in complex with proteins and other key biomolecules [178]. As accurately identifying such complex structures (in particular multi-ligand structures) can yield advanced insights into the binding dynamics and functional characteristics (and thereby, the medicinal potential) of numerous protein complexes *in vivo*, in recent years, significant resources have been spent developing new experimental and computational techniques for protein-ligand structure determination [179]. Over the last decade, machine learning (ML) methods for structure prediction have become indispensable components of modern structure determination at scale, with AlphaFold 2 for protein structure prediction being a hallmark example [15, 180].

As the field has gradually begun to investigate whether proteins in complex with other types of molecules can faithfully be modeled with ML (and particularly deep learning (DL)) techniques [150, 145, 30], several new works in this direction have

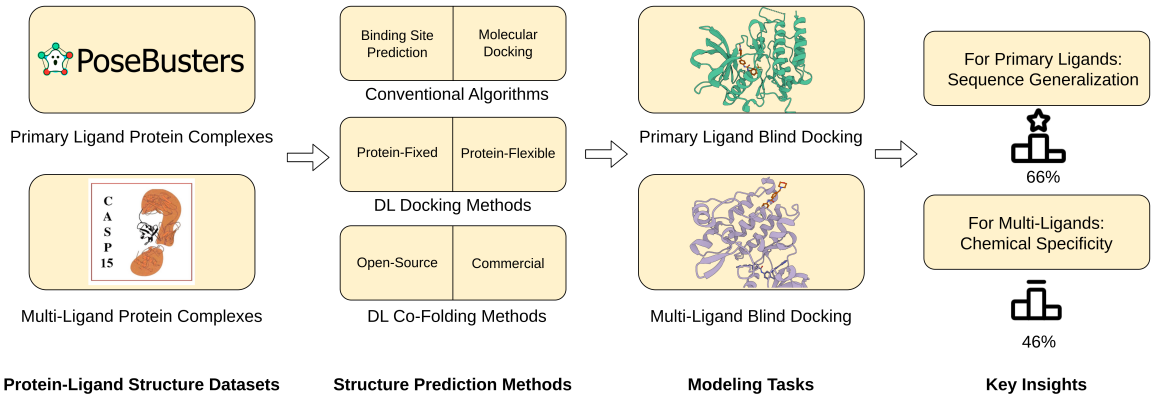



Figure 6.1: Overview of PoseBench, our comprehensive benchmark for *broadly applicable* DL modeling of primary and multi-ligand protein complex structures. Baseline methods of the benchmark include a range of the latest DL docking and co-folding methods, both open-source and commercially restrictive, as well as conventional algorithms for docking. Key observations derived using PoseBench include the discontinuity between structure and interaction modeling performance for novel or uncommon prediction targets and the heavy reliance of key DL co-folding methods on MSA-based input features to achieve high structural accuracy.

suggested the promising potential of such approaches to protein-ligand structure determination [99, 28, 29, 27]. Nonetheless, it remains to be shown the extent to which the latest of such DL methods can adequately generalize to the context of binding novel or uncommon protein-ligand interaction (PLI) pockets and multiple interacting ligand molecules (e.g., which can alter the chemical functions of various enzymes) as well as whether such methods can faithfully model amino acid-specific types of PLIs natively found in crystallized biomolecular structures.

To bridge this knowledge gap, our contributions in this work are as follows:

- We introduce the first unified benchmark for protein-ligand docking and structure prediction that evaluates the performance of several recent DL-based methods (e.g., AlphaFold 3, Chai-1) as well as conventional algorithms (e.g., AutoDock Vina) for primary and *multi*-ligand docking, which suggests that DL co-folding methods generally outperform conventional algorithms yet remain challenged by novel or uncommon prediction targets.

- In contrast to several recent works using crystal protein structures for protein-ligand docking [112, 181], the docking benchmark results we present in this work are all within the context of *standardized* input multiple sequence alignments (MSAs) and high accuracy *apo-like* (i.e., AlphaFold 3-predicted) protein structures without specifying known binding pockets, which notably enhances the broad applicability of this study's findings.

- Our newly proposed benchmark, PoseBench, enables specific insights into necessary areas of future work for accurate and generalizable biomolecular structure prediction, including that DL methods struggle to balance faithful modeling of native PLI fingerprints (PLIFs) with structural accuracy during pose prediction and that some DL co-folding methods are more dependent than others on the availability of input MSAs.

- Our benchmark results also highlight the importance of including challenging (out-of-sequence-distribution) datasets when evaluating future DL methods while measuring their ability to recapitulate amino acid-specific PLIFs with an appropriate new metric that we introduce in this work.

### 6.2.1 Related work

**Structure prediction of PLI complexes.** The field of DL-driven protein-ligand structure determination was largely sparked with the development of geometric deep learning methods such as EquiBind [108] and TANKBind [151] for direct (i.e., regression-based) prediction of bound ligand structures in protein complexes. Notably, these predictive methods could estimate localized ligand structures in complex with multiple protein chains as well as the associated complexes' binding affinities. However, in addition to their limited predictive accuracy, they have more recently been found to frequently produce steric clashes between protein and ligand atoms, notably hindering

their widespread adoption in modern drug discovery pipelines.

**Protein-ligand structure prediction and docking.** Shortly following the first wave of predictive methods for protein-ligand structure determination, DL methods such as DiffDock [99] demonstrated the utility of a new approach to this problem by reframing protein-ligand docking as a generative modeling task, whereby multiple ligand conformations can be generated for a particular protein target and rank-ordered using a predicted confidence score [182]. This approach has inspired many follow-up works offering alternative formulations of this generative approach to the problem [183, 184, 185, 186, 187, 28, 188, 189, 29, 190, 191, 181, 30, 31, 167, 192, 193, 27, 153, 194], with some of such follow-up works also being capable of accurately modeling protein flexibility upon ligand binding or predicting binding affinities to a high degree of accuracy.

**Benchmarking efforts for protein-ligand complexes.** In response to the large number of new methods that have been developed for protein-ligand structure prediction, recent works have introduced several new datasets and metrics with which to evaluate newly developed methods, with some of such benchmarking efforts focusing on modeling single-ligand protein interactions [195, 112, 176, 196, 197, 198, 199] and others specializing in the assessment of multi-ligand protein interactions [200]. One of the motivations for introducing PoseBench in this work is to bridge this gap by systematically assessing a selection of the latest (pocket-blind) structure prediction methods within both interaction regimes, using unbound (apo) protein structures with docking methods and challenging DL co-folding methods to predict full bioassemblies from primary sequences. As we will soon see, the benchmarking results in the following Section 6.3 demonstrate the relevance and utility of this comprehensive new evaluation suite for the future of protein-ligand modeling.

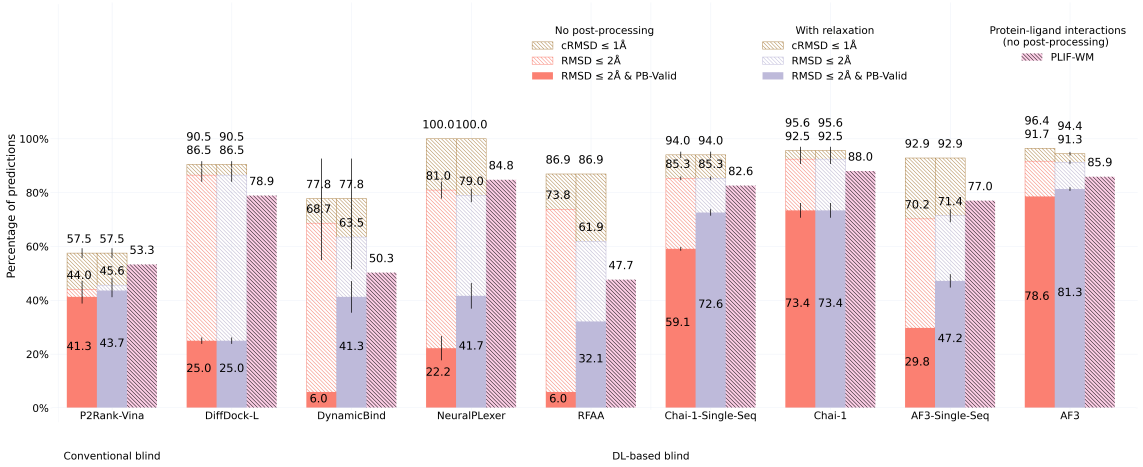



Figure 6.2: Astex Diverse primary ligand docking success rates (n=85).

## 6.3 RESULTS

In this section, we present PoseBench's results for primary and multi-ligand protein-ligand docking and structure prediction and discuss their implications for future work, as succinctly illustrated in Figure 6.1. Note that across all experiments, for generative methods, we report their performance metrics in terms of the mean and standard deviation across *three* independent runs of each method to gain insights into their inter-run stability and consistency. Key metrics include a method's percentage of structurally accurate ligand pose predictions with a (heavy atom centroid) root mean square deviation (RMSD) less than 2 (1) Å (i.e., (c)RMSD $\leq$ 2 (1) Å); its percentage of structurally accurate pose predictions that are also chemically valid according to the PoseBusters software suite (i.e., RMSD $\leq$ 2 Å & PB-Valid), which can be affected by the post-hoc application of structural relaxation driven by computationally expensive molecular dynamics (MD) simulations [201] (i.e., with relaxation); and our newly proposed Wasserstein matching score of its amino acid-specific predicted PLIFs (PLIF-WM). We formally define these metrics in Section 6.5.4. For interested readers, in Appendix E.3, we report the average runtime and memory usage of each baseline method to determine which methods are the most efficient for real-world structure-

based applications, and in Appendix E.7 we present supplementary results.

### 6.3.1 Astex Diverse results

Containing PLI structures deposited in the RCSB Protein Data Bank (PDB) [173] up until 2007, most of the well-known Astex Diverse dataset's structures [177] are present in the training data of each baseline method, yet benchmarking results for this dataset (n=85), shown in Figure 6.2, indicate that only DL co-folding methods achieve higher structural and chemical accuracy rates (RMSD $\leq$ 2 Å & PB-Valid) than the conventional docking baseline AutoDock Vina combined with P2Rank for PLI binding site prediction to facilitate blind molecular docking. Interestingly, nearly all baseline methods identify the correct PLI binding pocket approximately 90% of the time, yet only the DL co-folding methods AlphaFold 3 (AF3) [27] and Chai-1 [153] achieve a reasonable balance between their rates of structural and chemical accuracy and chemical specificity (PLIF-WM), with the single-sequence (i.e., MSA-ablated) version of AF3 being a notable exception. These results suggest that DL co-folding methods have learned the most comprehensive representations of this dataset's input sequences, yet only the performance of the DL co-folding method Chai-1 is maintained without the availability of diverse input MSAs. One likely explanation for this phenomenon is that Chai-1's training primarily relied upon the availability of amino acid sequence embeddings generated by the protein language model ESM2 [25] in addition to features derived from input MSAs, which may have imbued the model with rich MSA-*independent* representations for biomolecular structure prediction.

### 6.3.2 DockGen-E results

As visualized in Figure 6.3, results with our new DockGen-E dataset of biologically relevant PLI complexes deposited in the PDB up to 2019 (n=122) demonstrate that only the latest DL co-folding methods can locate a sizable fraction of structurally

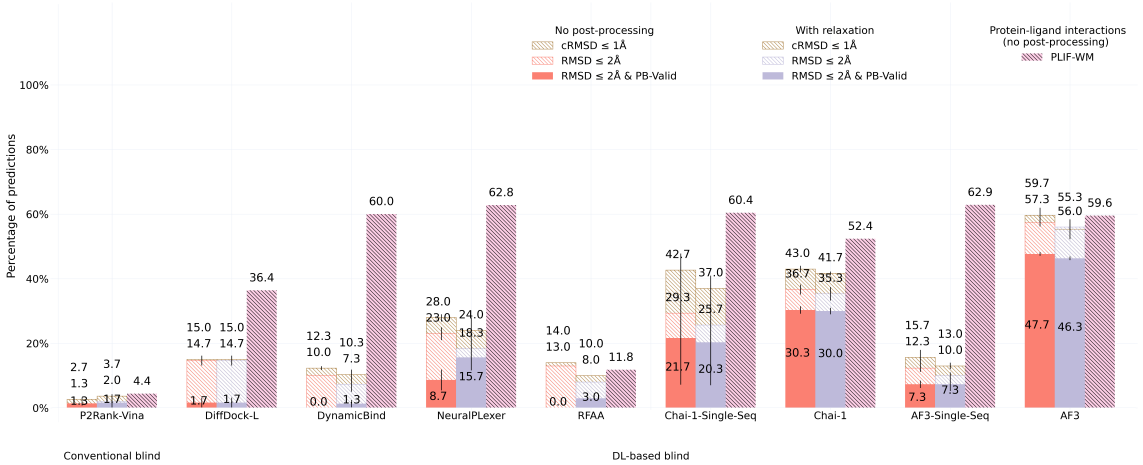



Figure 6.3: DockGen-E primary ligand docking success rates (n=122).

accurate PLI binding poses represented in this dataset. As such methods may have previously seen these PLI structures in their respective training data, it is surprising that even the latest AF3 model fails to identify a structurally and chemically accurate pose for more than half of the dataset's complexes. Further, for Chai-1 and AF3, their single-sequence variants achieve slightly higher chemical specificity than their MSA-based versions, which may indicate that for these methods MSA features obfuscate primary sequence knowledge in favor of evolution-averaged (i.e., amino acid-generic) representations. The overall lower range of PLIF-WM values achieved by each method for this dataset further suggests the increased chemical modeling difficulty of this dataset's complexes compared to those presented by the Astex Diverse dataset. A potential source of these difficulties is that each of this dataset's complexes represents a functionally distinct PLI binding pocket (as codified by ECOD domains [202], see [181] for more details) compared to data deposited in the PDB before 2019. As such, it is likely that AF3 and Chai-1 are "overfitted" to the most common types of PLI structures in the PDB and may overlook several uncommon types of PLI binding pockets present in nature.

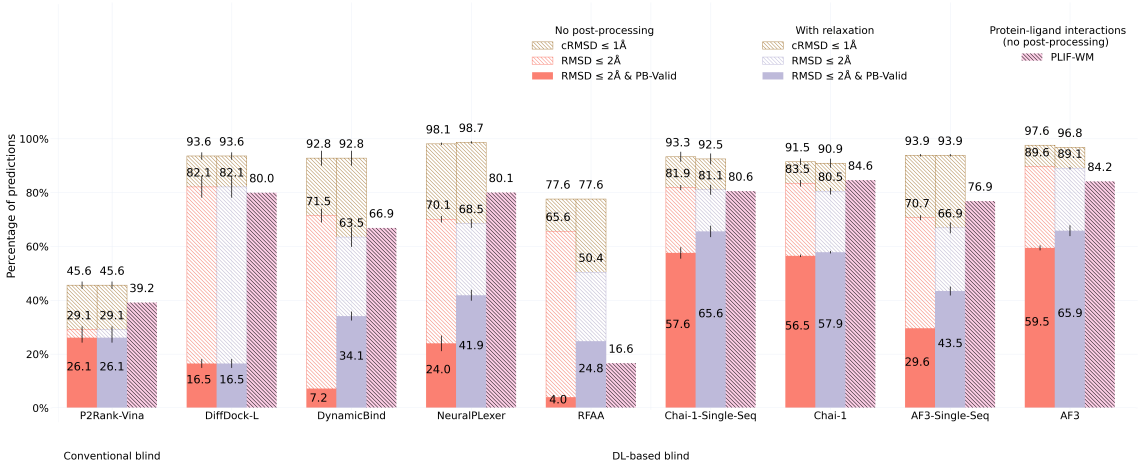



Figure 6.4: PoseBusters Benchmark primary ligand docking success rates (n=130/308).

### 6.3.3 PoseBusters Benchmark results

With approximately half of its PLI structures deposited in the PDB after AF3's maximum-possible training data cutoff of September 30, 2021 (n=308 total, filtered to n=130 for subsequent analyses), the PoseBusters Benchmark dataset's results, presented in Figure 6.4, indicate once again that DL co-folding methods achieve top performance compared to conventional and DL docking baseline methods. Nonetheless, we observe an interesting phenomenon whereby Chai-1 strikes a balance of structural and chemical accuracy and chemical specificity comparable to AF3 even without input MSAs, potentially suggesting that Chai-1 achieves stronger sequence generalization for this dataset than AF3. Moreover, with the single-sequence version of AF3, we again observe significant degradations in its overall performance, whereas running Chai-1 with input MSAs achieves higher chemical specificity at the cost of marginal structural accuracy compared to running it in single-sequence mode. These observations highlight the importance in future work of carefully studying why and how the *training* of biomolecular structure generative models can be influenced to varying degrees by the availability and composition of diverse input MSAs.

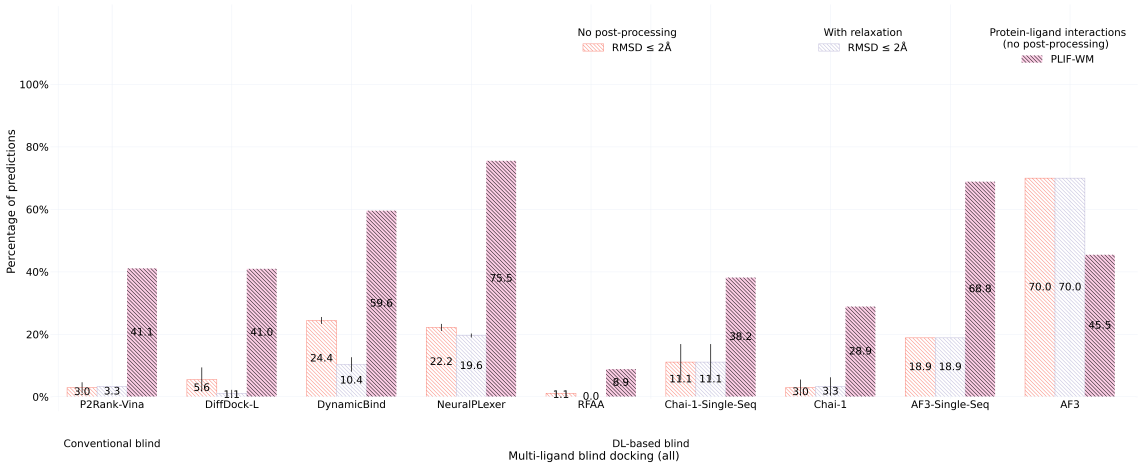



Figure 6.5: CASP15 multi-ligand docking success rates (n=13).

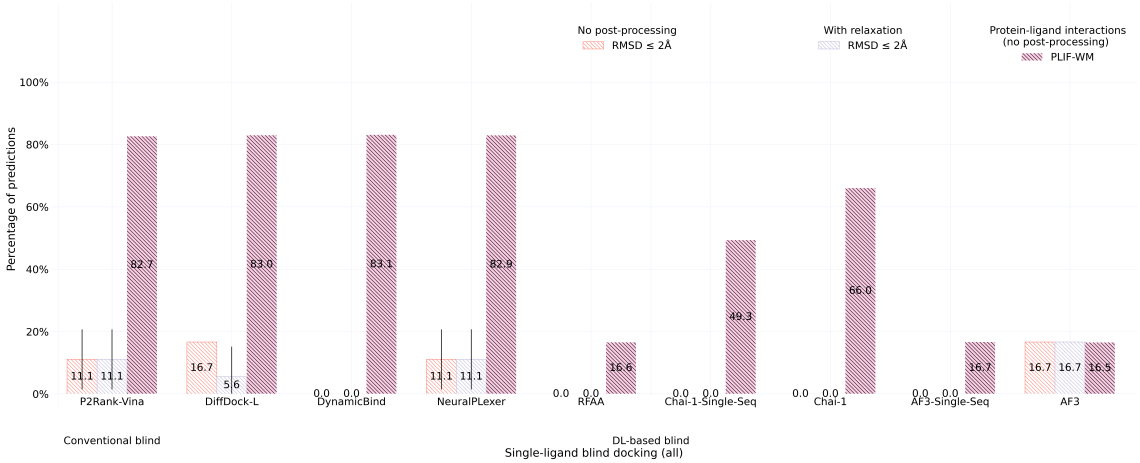



Figure 6.6: CASP15 single-ligand docking success rates (n=6).

### 6.3.4 CASP15 results

As a new dataset of novel and challenging PLI complexes on which no method has been trained, the CASP15 dataset's multi-ligand results (n=13), illustrated in Figure 6.5, indicate that most methods fail to adequately generalize to multi-ligand prediction targets, yet AF3 stands out in this regard (only) when provided input MSAs. As many of these CASP15 multi-ligand targets represent large, highly symmetric protein complexes, it is likely that additional evolutionary information in the form of MSAs has improved AF3's ability to predict higher-order protein-protein interactions for

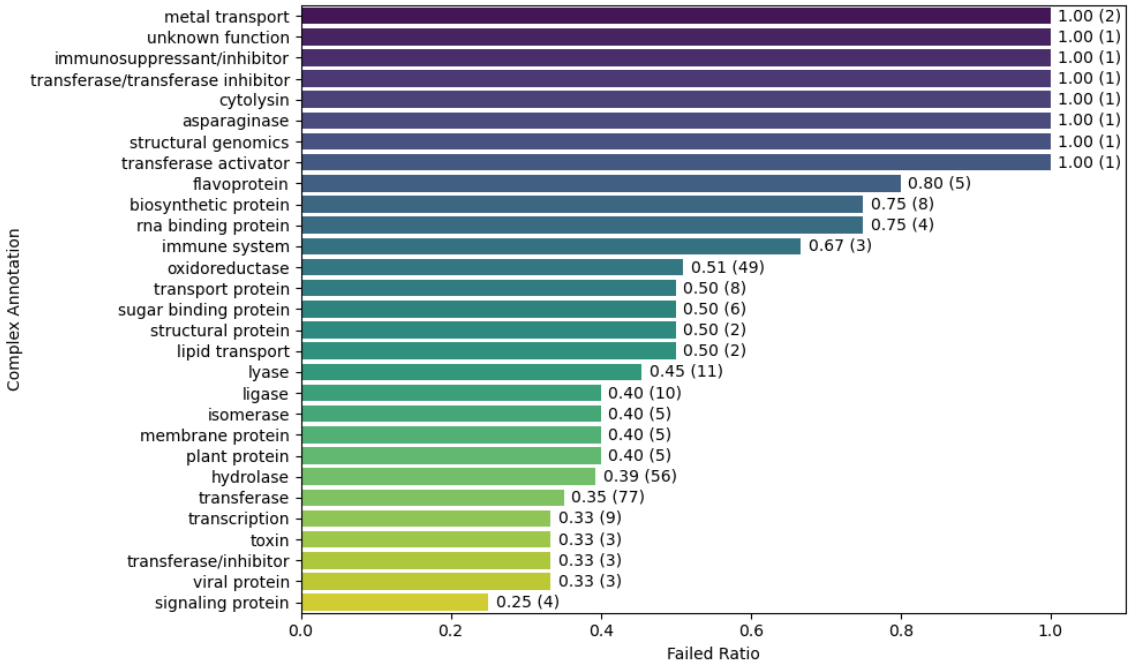



Figure 6.7: Function annotations of the PLI complexes all methods mispredicted (n=129).

these targets, yet interestingly its improved rate of structural accuracy comes at the cost of its protein-*ligand* chemical specificity (in comparison to its single-sequence results). For the CASP15 dataset's single-ligand (i.e., primary ligand) results (n=6) presented in Figure 6.6, this trend is subverted in that conventional docking and simpler DL co-folding methods such as AutoDock Vina and NeuralPLexer outperform all recent DL co-folding methods in modeling crystalized PLIFs while achieving comparable rates of structural accuracy. Given the small size of the CASP15 dataset, it is reasonable to conclude that DL methods, in particular the latest co-folding methods, *may* be challenged to predict PLI complexes containing novel PLIs mediated by novel protein sequences. In the following Section 6.3.5, we will explore this latter point in greater detail by analyzing the protein sequence similarities between common PDB training data and this benchmark's evaluation datasets.

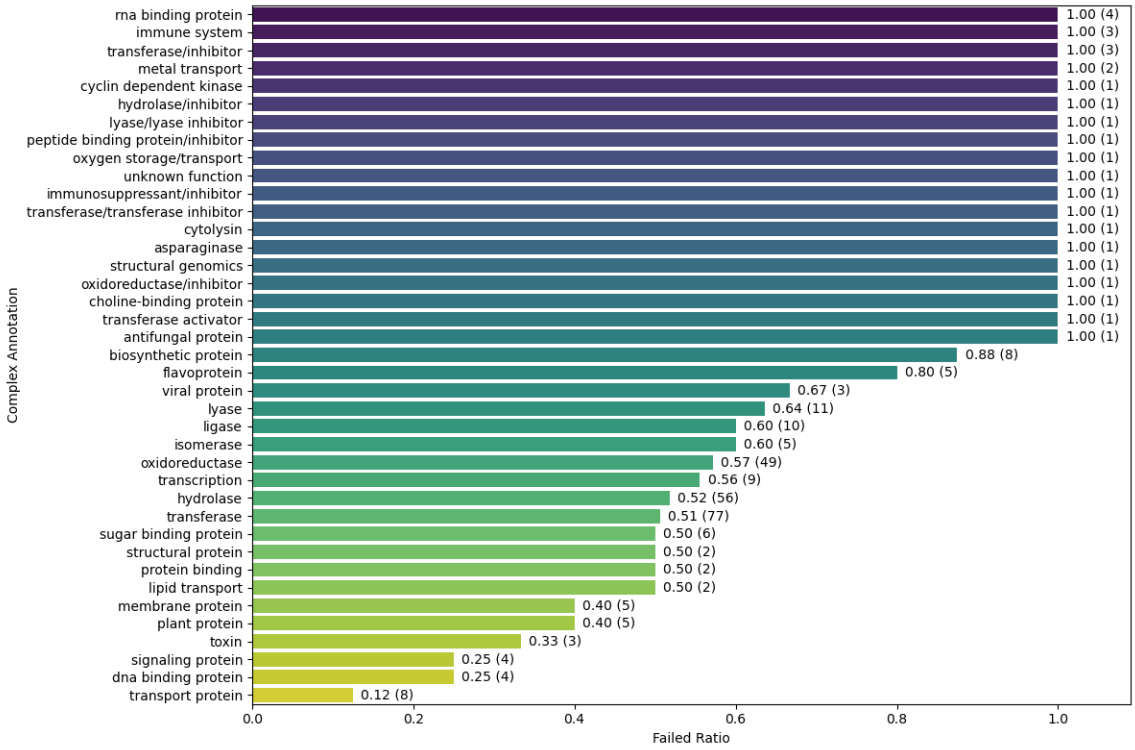



Figure 6.8: Function annotations of the PLI complexes AF3 mispredicted (n=171).

### 6.3.5 Exploratory analyses of results

In this section, we explore a range of questions to study the common "failure" modes of the baseline methods included in this work, to outline new directions for future research and development efforts in drug discovery.

**Research Question 1**: What are the most common types of protein-ligand complexes that *all* baseline methods fail to predict?

→ To address this query, we first collect all ligand pose predictions that no method could predict with structural and chemical accuracy (according to the metric RMSD $\leq$ 2 Å & PB-Valid). For each of these "failed" ligand poses, we retrieve the PDB's functional annotation of the protein in complex with this ligand and construct a histogram to visualize the frequency of these (failed complex) annotations. The results of this analysis are presented in Figure 6.7, in which we see that metal transport

proteins, flavoproteins, biosynthetic proteins, RNA binding proteins, immune system proteins, and oxidoreductases are commonly mispredicted by all baseline methods such as Chai-1 and RoseTTAFold-All-Atom (RFAA) [30], suggesting these classes of proteins may be largely unaddressed by the most recent DL methods for PLI structure prediction. To illuminate potential future research directions, in the next analysis, we investigate whether this pattern persists specifically for one of the latest DL *co-folding* methods, AF3.

**Research Question 2**: What are the most common types of protein-ligand complexes that DL *co-folding* methods such as AF3 fail to predict?

→ For this follow-up question, we link all of AF3's failed ligand predictions with corresponding protein function annotations available in the PDB to understand which types of PLI complexes AF3 finds the most difficult to predict. Similar to the answer to our first research question, Figure 6.8 shows that, in order of difficulty, AF3 is most challenged to produce ligand poses of high structural and chemical accuracy for ligand-bound RNA binding proteins, immune system proteins, metal transport proteins, biosynthetic proteins, flavoproteins, lyases, and oxidoreductases. As several of these classes of proteins have not been well represented in the PDB over the last 50 years (e.g., immune system and biosynthetic proteins), in future work, it will be important to ensure that either the performance of new DL methods for PLI structure prediction is expanded to support accurate modeling of these uncommon types of ligand-bound proteins or a broadly applicable fine-tuning method for uncommon types of interactions is proposed.

**Research Question 3**: How often is *lack* of sequence homology to PDB training data associated with failed predictions by DL co-folding methods such as AF3?

→ To understand the impact of protein sequence similarity on the performance

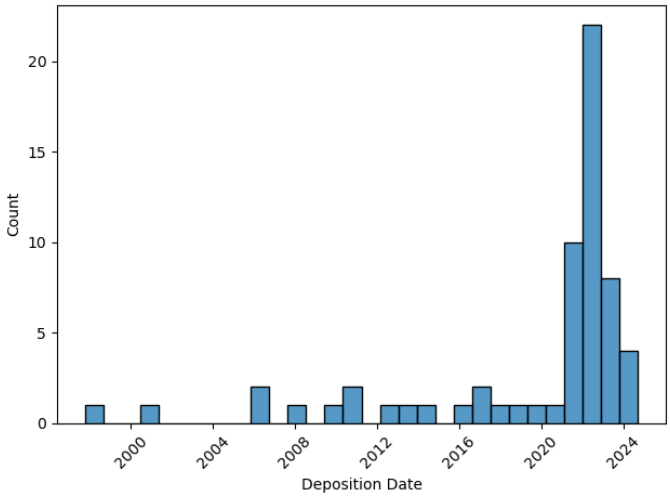
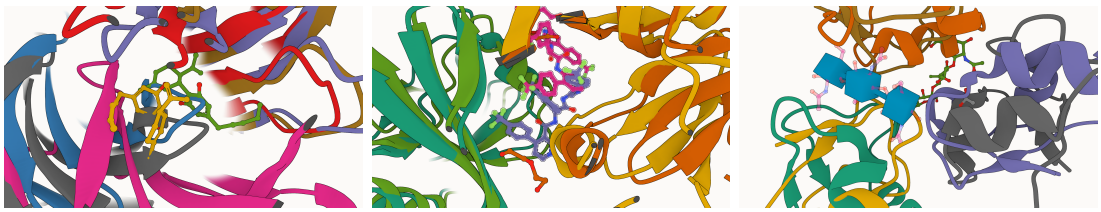



Figure 6.9: Sequence homologs of the unseen PLI complexes AF3 mispredicted (n=62).

of the DL co-folding method AF3, we isolate the subset of failed ligand pose predictions AF3 made for the PoseBusters Benchmark and CASP15 datasets, as none of these datasets' prediction targets are contained in AF3's training dataset. We then use MMseqs2 [203] to identify the deposition dates of the most similar (i.e., top hit) protein chains with 30% or greater sequence homology to any protein chain in the unseen PLI complexes AF3 failed to predict. Figure 6.9 reveals that most of the unseen PLI complexes AF3 failed to predict were not associated with *any* protein sequence homologs present in its PDB-based training dataset. That is, when AF3 failed to predict a new PLI complex, it also could not rely on sequence homology to its training dataset to bolster its performance. This observation suggests that the performance of recent DL co-folding methods for novel protein sequences or PLI complexes may be limited by the extent to which the method can "retrieve" similar sequence representations from its training data. We conclude our quantitative analyses with an illustration of the different failure modes of each baseline method, as depicted in Figure 6.10.



(a) Biosynthetics (RFAA) (b) Immune Proteins (AF3) (c) Novel Proteins (AF3)

Figure 6.10: Examples of baseline methods' three failure modes discovered using PoseBench.

## 6.4 DISCUSSION

In this chapter, we have introduced PoseBench, a unified, broadly applicable benchmark and toolkit for studying the performance of DL methods for protein-ligand docking and structure prediction. Benchmarking results with PoseBench suggest that DL co-folding methods generally outperform conventional and DL docking baselines yet remain challenged to predict protein targets containing novel sequences. Further, we find that several DL methods face difficulties balancing the structural accuracy of their predicted poses with the chemical specificity of their induced protein-ligand interactions, highlighting that future methods may benefit from the introduction of *physico-chemical* loss functions or sampling techniques to bridge this performance gap. Lastly, we observe that some (but not all) DL co-folding methods are highly dependent on the availability of diverse input MSAs to achieve high structural prediction accuracy, underscoring the need in future work to elucidate the impact of the availability of MSAs and protein language model embeddings on the training dynamics of biomolecular structure prediction methods. As a publicly available resource, PoseBench is flexible to accommodate new datasets and methods for protein-ligand docking and structure prediction.

Table 6.1: PoseBench evaluation datasets of protein-(multi-)ligand structures.

| Name | Type | Source | Size (Total # Ligands) |
|---|---|---|---|
| Astex Diverse | Primary Ligand | [177] | 85 |
| PoseBusters Benchmark | Primary Ligand | [112] | 130/308 |
| DockGen-E | Primary Ligand | | 122 |
| CASP15 | Multi-Ligand | | 102 (across 19 complexes)<br>→ 6 (13) single (multi)-ligand complexes |

## 6.5 METHODS

### 6.5.1 PoseBench

The overall goal of PoseBench, our newly designed benchmark for protein-ligand docking and structure prediction, is to provide the research community with a centralized resource with which one can systematically measure, in a variety of macromolecular contexts, the methodological advancements of new conventional and DL methods proposed for this domain. In the following sections, we describe PoseBench's design and composition (as portrayed in Figure 6.1) and how we have used PoseBench to evaluate several recent DL docking and co-folding methods (as well as a strong conventional baseline algorithm) for protein-ligand structure modeling.

### 6.5.2 Benchmark datasets

As shown in Table 6.1, PoseBench provides users with *broadly applicable*, preprocessed versions of four datasets with which to evaluate existing or new protein-ligand structure prediction methods: Astex Diverse [177], PoseBusters Benchmark [112], and the new DockGen-E and CASP15 PLI datasets that we have manually curated in this work.

**Astex Diverse dataset.** The Astex Diverse dataset is a collection of 85 PLI complexes composed of various drug-like molecules and cofactors known to be of pharmaceutical or agrochemical interest, where a primary (representative) ligand is

annotated for each complex. This dataset can be considered an easy benchmarking dataset for methods trained on recent data contained in the PDB in that most of its complexes (deposited in the PDB up to 2007) are known to overlap with the commonly used PDBBind 2020 (time-split) training dataset [204, 108] containing complexes deposited in the PDB before 2019. As such, including this dataset for benchmarking allows one to estimate the *breadth* of a method's structure prediction capabilities for important primary ligand protein complexes represented in the PDB.

To perform unbound (apo) protein-ligand docking with this dataset, we used AF3 to predict the structure of each of its protein complexes, with all ligands and cofactors excluded. We then optimally aligned these predicted protein structures to the corresponding crystal (holo) PLI complex structures using a PLI binding site-focused structural alignment performed using PyMOL [205], where each binding site is defined as all amino acid residues containing crystallized heavy atoms that are within 10 Å of any crystallized ligand heavy atom. To enable the broad availability of PoseBench's benchmark datasets in both commercial and academic settings, we also provide unbound (apo) protein structures predicted using the MIT-licensed ESMFold model [25], although in Section 6.3 we report results using AF3's predicted structures as the default data source. We further note that on average across all benchmark datasets and methods, AF3's predicted structures improve baseline docking methods' structural accuracy rates by 5-10%.

**PoseBusters Benchmark dataset.** Version 2 of the the popular PoseBusters Benchmark dataset [112], which we adopt in this work, contains 308 recent primary ligand protein complexes deposited in the PDB from 2019 onwards. Accordingly, in contrast to Astex Diverse, this dataset can be considered a moderately difficult benchmark dataset for baseline methods, since many of its complexes do not directly overlap with the most commonly used PDB-based training data. Important to note is that, among all baseline methods, AF3 used the most recent PDB training data cutoff

of September 30, 2021, which motivated us to report the results in Section 6.3.3 for only the subset of PoseBusters Benchmark complexes (n=130) deposited in the PDB after this date. Like Astex Diverse, for the PoseBusters Benchmark dataset, we used AF3 (and ESMFold) to predict the *apo* protein structures of each of its complexes and then performed our PyMOL-based structural binding site alignments.

**DockGen-E dataset.** The original DockGen dataset [181] contains 189 diverse primary ligand protein complexes, each representing a functionally distinct type of PLI binding pocket according to ECOD domain partitioning [202, 181]. Consequently, this dataset can be considered PoseBench's most difficult primary ligand dataset to model since its PLI binding sites are distinctly uncommon compared to those frequently found in the training datasets of all baseline methods, though it is important to note that these original DockGen complexes were deposited in the PDB from 2019 onward, making this benchmarking dataset partially overlap with the training datasets of baseline DL co-folding methods such as AF3 and Chai-1. Nonetheless, in line with our initial hypotheses, the benchmarking results in Section 6.3 demonstrate that no baseline method can adequately predict the PLI binding sites and ligand poses represented by this bespoke subset of the PDB, suggesting that *all* baseline DL methods have yet to learn *broadly applicable* representations of protein-ligand binding.

Unfortunately, the original DockGen dataset contains only the primary protein chains representing each novel binding pocket after filtering out all non-interacting chains and cofactors in a given biological assembly (bioassembly), which considerably *reduces* the biophysical context provided to baseline methods to make reasonable predictions. As such, we argue for the need to construct a new dataset that challenges baseline methods (in particular DL co-folding methods) to predict full bioassemblies containing novel PLI binding pockets, which we address with our enhanced version of DockGen called DockGen-E.

To construct DockGen-E, we collected the original DockGen dataset's PLI binding pocket annotations for each complex. We then retrieved the corresponding first bioassembly listed in the PDB to obtain each PDB entry's biologically *relevant* complex, filtering out DockGen complexes for which the first bioassembly could not be mapped to its original PLI binding pocket annotation (which indicates these original DockGen PLI binding pockets were initially not derived from the PDB's corresponding first bioassembly). This procedure left 122 biologically relevant assemblies remaining for benchmarking. Like Astex Diverse and PoseBusters Benchmark, for DockGen-E, we then used AF3 (and ESMFold) to predict the unbound (apo) protein structures of each complex in the dataset and structurally aligned the predicted protein structures to their corresponding crystallized PLI binding sites using PyMOL.

**CASP15 dataset.** To assess the *multi*-primary ligand (i.e., multi-ligand) modeling capabilities of recent methods for protein-ligand docking and structure prediction, with PoseBench, we introduce a preprocessed, DL-ready version of the CASP15 PLI dataset debuted as a first-of-its-kind prediction category in the 15th Critical Assessment of Techniques for Structure Prediction (CASP) competition held in 2022 [200]. The CASP15 PLI dataset is originally comprised of 23 protein-ligand complexes released in the PDB from 2022 onward, where we subsequently filter out 4 complexes based on (1) whether the CASP organizers ultimately assessed predictions for the complex and (2) whether they are nucleic acid-ligand complexes with no interacting protein chains. The 19 remaining PLI complexes, which contain a total of 102 (fragment) ligands, consist of a variety of ligand types including single-atom (metal) ions and large drug-sized molecules with up to 92 atoms in each (fragment) ligand. As such, this dataset is appropriate for assessing how well structure prediction methods can model interactions between different (fragment) ligands in the same complex, which can yield insights into the inter-ligand steric clash rates of each method. As with all other benchmark datasets, we used AF3 (and ESMFold) to predict the un-

bound (apo) structure of each protein complex in the dataset and then performed a PyMOL-based structural alignment of the corresponding PLI binding sites.

**PLI similarity analysis between datasets.** For an investigation of the similarity of PLIs represented in each dataset, in Appendix E.5, we analyze the different types and frequencies of common, ProLIF-annotated protein-ligand binding pocket interactions [206] natively found within the common PDBBind 2020 training dataset and the Astex Diverse, PoseBusters Benchmark, DockGen-E, and CASP15 datasets, respectively, to quantify the diversity of the (predicted) interactions each dataset can be used to evaluate. In short, we find that the DockGen-E and CASP15 benchmark datasets are the *most dissimilar* compared to the common PDBBind 2020 training dataset, further illustrating the unique PLI modeling challenges offered by these evaluation datasets.

### 6.5.3 Formulated tasks

In this work, we developed PoseBench to focus our analysis on the behavior of different conventional and DL methods for protein-ligand structure prediction in a variety of macromolecular contexts (e.g., with or without inorganic cofactors present). With this goal in mind, below we formalize the structure prediction tasks currently available with PoseBench, with its source code flexibly designed to accommodate new tasks in future work.

**Primary ligand blind docking.** For primary ligand blind docking, each baseline method is provided with a complex's (multi-chain) protein sequence and an optional predicted (apo) protein structure as input along with its corresponding (fragment) ligand SMILES strings, where fragment ligands include the *primary* binding ligand to be scored as well as all cofactors present in the corresponding crystal structure. In particular, no knowledge of the complex's PLI binding pocket is provided to evaluate how well each method can (1) identify the correct PLI binding pockets and (2)

correct ligand poses within each pocket (3) with high chemical validity and (4) specificity for the pockets' amino acid residues. After all fragment ligands are predicted, PoseBench extracts each method's prediction of the primary binding ligand and reports evaluation results for these primary predictions.

**Multi-ligand blind docking.** For multi-ligand blind docking, each baseline method is provided with a complex's (multi-chain) protein sequence and an optional predicted (apo) protein structure as input along with its corresponding (fragment) ligand SMILES strings. As in primary ligand blind docking, no knowledge of the PLI binding pockets is provided, which offers the opportunity to evaluate not only PLI binding pocket and conformation prediction accuracy but, in the context of multi-binding ligands, also inter-ligand steric clash rates.

### 6.5.4 Metrics

**Traditional metrics.** For PoseBench, we reference two key metrics in the field of structural bioinformatics: the root-mean-square deviation (RMSD) and local Distance Difference Test (lDDT) [207]. The RMSD between a predicted 3D conformation (with atomic positions $\hat{x}_i$ for each of the molecule's $n$ heavy atoms) and the ground-truth (crystal structure) conformation ($x_i$) is defined as:

$$\text{RMSD} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\|\hat{x}_i - x_i\|^2}. \tag{6.1}$$

The lDDT score, which is commonly used to compare predicted and ground-truth protein 3D structures, is defined as:

$$\text{lDDT} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{4}\sum_{k=1}^{4}\left(\frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}\Theta(|\hat{d}_{ij} - d_{ij}| < \Delta_k)\right), \tag{6.2}$$

where $N$ is the total number of heavy atoms in the ground-truth structure; $\mathcal{N}_i$ is the set of neighboring atoms of atom $i$ within the inclusion radius $R_o = 15$ $\mathring{A}$ in

the ground-truth structure, excluding atoms from the same residue; $\hat{d}_{ij}$ ($d_{ij}$) is the distance between atoms $i$ and $j$ in the predicted (ground-truth) structure; $\Delta_k$ are the distance tolerance thresholds (i.e., 0.5 Å, 1 Å, 2 Å, and 4 Å); $\Theta(x)$ is a step function that equals 1 if x is true, and 0 otherwise; and $|\mathcal{N}_i|$ is the number of neighboring atoms for atom $i$. As originally proposed by [200], in this study, we adopt the PLI-specific variant of lDDT for scoring *multi*-ligand complexes, which calculates lDDT scores to compare predicted and ground-truth protein-(multi-)ligand complex structures following optimal (chain-wise and residue-wise) structural alignment of the predicted and ground-truth PLI binding pockets.

Lastly, we also measure the molecule validity rates of each predicted PLI complex pose using the PoseBusters software suite (i.e., PB-Valid) [112]. This suite runs several important chemical and structural sanity checks for each predicted pose including energy ratio inspection and geometric (e.g., flat ring) assertions which provide a secondary filter of accurate poses that are also chemically and structurally meaningful.

**New metrics.** The RMSD, lDDT, and PB-Valid metrics of a protein-ligand binding structure provide useful characterizations of how accurate and reasonable a predicted pose is. However, a key limitation of these metrics is that they do not measure how well a predicted pose resembles a native pose when comparing their induced PLIFs. Recently, [196] introduced a complementary benchmarking metric, PLIF-valid, assessing DL methods' recovery rates of known PLIs. However, this metric only reports a strict recall rate of each method's interaction types rather than a continuous measure of how well each method's interactions match the *distribution* of crystalized PLIs. Moreover, in drug discovery, a primary concern when designing new drug candidates is ensuring they produce *amino acid-specific* types of interactions (and not others), hence we desire each baseline method to recall the correct types of PLIs for each pose and to avoid predicting (i.e., hallucinating) types of interactions that are not natively present. Consequently, we argue that an ideal PLI-aware

benchmarking metric is a single continuous metric that assesses the recall and precision of a method's predicted *distribution* of *amino acid-specific* PLIFs. To this end, we propose two new benchmarking metrics, PLIF-EMD and PLIF-WM.

For each PLI complex, PLIF-EMD measures the Earth mover's distance (EMD) [208] between a method's predicted histogram of PLI type counts $u$ (specific to each type of interaction) and the corresponding native histogram $v$, where each histogram of interaction type counts is represented as a 1D discrete distribution. Formally, this equates to computing the Wasserstein distance between these two 1D distributions $u$ and $v$ as

$$\text{PLIF-EMD} \;:=\; l_1(u,v) = \inf_{\pi \in \Pi(u,v)} \int_{\mathbb{R}\times\mathbb{R}} |x-y| d\pi(x,y), \tag{6.3}$$

where $\Pi(u,v)$ denotes the set of distributions on $\mathbb{R}\times\mathbb{R}$ whose marginals, $u$ and $v$, are on the first and second factors, respectively. To penalize a baseline method for producing non-native interaction types, we unify the bins in each histogram before converting them into 1D discrete representations. Namely, to perform this calculation, each PLI is first represented as a fingerprint tuple of <ligand type, amino acid type, interaction type> as determined by the software tool ProLIF [206] and then grouped to count each type of tuple to form a histogram. As such, a lower PLIF-EMD value implies a better continuous agreement between predicted and native interaction histograms. PLIF-WM, derived from PLIF-EMD, assesses the Wasserstein matching (WM) score of a pair of PLIF histograms. Specifically, to obtain a more benchmarking-friendly score ranging from 0 to 1 (higher is better), we define PLIF-WM as

$$\text{PLIF-WM} \;:=\; 1 - \frac{\text{PLIF-EMD} - \text{PLIF-EMD}_{\text{min}}}{\text{PLIF-EMD}_{\text{max}} - \text{PLIF-EMD}_{\text{min}}}, \tag{6.4}$$

where $\text{PLIF-EMD}_{\text{min}}$ and $\text{PLIF-EMD}_{\text{max}}$ denote the minimum (best) and maximum (worst) values of PLIF-EMD, respectively. As a metric normalized relative to each collection of the latest baseline methods, PLIF-WM allows one to quickly identify

which of the latest methods has the greatest capacity to produce realistic distributions of PLIs. As a practical note, we use SciPy 1.15.1 [209] to provide users of PoseBench with an optimized implementation of PLIF-EMD and thereby PLIF-WM.

### 6.5.5 Baseline methods and experimental setup

**Overview.** We designed PoseBench to answer specific modeling questions for PLI complexes such as (1) which types of methods (if any) can predict both common and uncommon PLI complexes with high structural and chemical accuracy and (2) which most accurately predict multi-ligand structures without steric clashes? In the following sections, we discuss which types of methods we evaluate in our benchmark and how we evaluate each method's predictions for PLI complex targets.

**Method categories.** As illustrated in Figure 6.1, to explore a range of the most well-known or recent methods to date, we divide PoseBench's baseline methods into one of three categories: (1) conventional algorithms, (2) DL docking algorithms, and (3) DL co-folding algorithms.

As a representative algorithm for conventional protein-ligand docking, we pair AutoDock Vina (v1.2.5) [210] for molecular docking with P2Rank for protein-ligand binding site prediction [211] to form a strong conventional (blind) docking baseline (P2Rank-Vina) for comparison with DL methods. To represent DL docking methods, we include DiffDock-L [181] for docking with static protein structures and DynamicBind [28] for flexible docking. Lastly, to represent some of the latest DL co-folding methods, we include NeuralPLexer [29], RFAA [30], AF3 [27], and Chai-1 [153]. For interested readers, each method's input and output data formats are described in Appendix E.6.

**Prediction and evaluation procedures.** The PLI complex structures each method predicts are subsequently evaluated using different structural and chemical accuracy and molecule validity metrics depending on whether the targets are pri-

mary or multi-ligand complexes. In Section 6.5.4, we provide formal definitions of PoseBench's evaluation metrics. Note that if a method's prediction raises any errors in subsequent scoring stages (e.g., due to missing entities or formatting violations), the prediction is excluded from the evaluation.

**Primary ligand evaluation.** For primary ligand targets, we report each method's percentage of (top-1) ligand conformations within 2 Å of the corresponding crystal ligand structure (RMSD $\leq$ 2 Å), using 1 Å to instead assess whether the predicted ligand's heavy atom centroid (i.e., binding pocket) was correct (cRMSD $\leq$ 1 Å), as well as the percentage of such "correct" ligand conformations that are also considered to be chemically and structurally valid according to the PoseBusters software suite [112] (RMSD $\leq$ 2 Å & PB-Valid). Importantly, as described in Section 6.5.4, we also report each method's new PLIF-WM scores to study the relationship between its structural accuracy and chemical specificity.

**Multi-ligand evaluation.** Similar to the protein-ligand scoring procedure employed in the CASP15 competition [200], for multi-ligand targets, we report each method's (top-1) percentage of "correct" (binding site-superimposed) ligand conformations (RMSD $\leq$ 2 Å) as well as violin plots of the RMSD and PLI-specific lDDT scores of its protein-ligand conformations across all (fragment) ligands within the benchmark's multi-ligand complexes (see Appendix E.7 for these plots). Notably, this latter metric, referred to as lDDT-PLI, allows one to evaluate specifically how well each method can model protein-ligand structural interfaces. Additionally, we report each method's PB-Valid rates (calculated once for each multi-ligand complex) and PLIF-WM scores.

# Chapter 7

# PROTEIN-LIGAND STRUCTURE AND AFFINITY PREDICTION IN CASP16 USING A GEOMETRIC DEEP LEARNING ENSEMBLE AND FLOW MATCHING

Adapted from Alex Morehead, Jian Liu, Pawan Neupane, Nabin Giri, and Jianlin Cheng. "Protein-ligand structure and affinity prediction in CASP16 using a geometric deep learning ensemble and flow matching". *CASP16-invited issue of Proteins: Structure, Function, and Bioinformatics* (2025).

## 7.1 ABSTRACT

Predicting the structure of ligands bound to proteins is a foundational problem in modern biotechnology and drug discovery, yet little is known about how to combine the predictions of protein-ligand structure (poses) produced by the latest deep learning methods to identify the best poses and how to accurately estimate the binding affinity between a protein target and a list of ligand candidates. Further, a blind benchmarking and assessment of protein-ligand structure and binding affinity prediction is necessary to ensure it generalizes well to new settings. Towards this end, in this chapter, we introduce MULTICOM_LIGAND, a deep learning-based protein-ligand structure and binding affinity prediction ensemble featuring structural consensus ranking for unsupervised pose ranking and a new deep generative flow matching model for joint structure and binding affinity prediction. Notably, MULTICOM_LIGAND

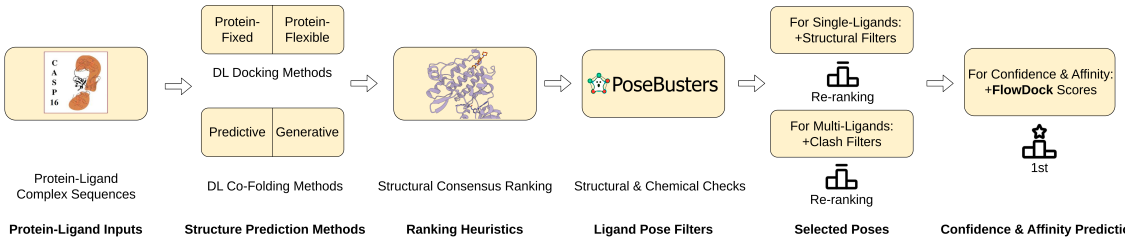



Figure 7.1: A high-level overview of MULTICOM_LIGAND, our proposed ensembling method for protein-ligand structure and binding affinity prediction. Given the sequence of a single- or multi-chain protein and the SMILES string of one or multiple ligands, MULTICOM_LIGAND predicts a number of plausible ligand poses using a selection of the latest deep learning (DL) methods for protein-ligand modeling. Such poses are then rank-ordered using an unsupervised structural consensus ranking heuristic that biases towards the most structurally similar poses across all methods and further filtered using a variety of structural and chemical sanity checks provided by the PoseBusters software suite. Finally, the top-ranked poses are evaluated by our new deep generative flow model FLOWDOCK for joint assessment of protein-ligand structure confidence scores and binding affinities.

ranked among the top-5 ligand prediction methods in both protein-ligand structure prediction and binding affinity prediction in the 16th Critical Assessment of Techniques for Structure Prediction (CASP16), demonstrating its efficacy and utility for real-world drug discovery efforts. The source code for MULTICOM_LIGAND is freely available at `https://github.com/BioinfoMachineLearning/MULTICOM_ligand`.

## 7.2 INTRODUCTION

The effects of ligands binding to proteins are numerous and foundational to research efforts in biotechnology and drug discovery, yet efficiently determining the structure, and thereby the function, of such ligand-bound protein complexes has challenged the structural biology community for decades. In the 15th Critical Assessment of Techniques for Structure Prediction (CASP15) [200], template-based approaches to protein-ligand structure determination generally outperformed those based on deep learning (DL). Nonetheless, over the last few years, several new DL methods (mostly diffusion models [212]) for protein-ligand docking and structure prediction have been

introduced [213, 28, 170, 30, 29], importantly raising the question of which method(s) perform(s) best for a range of diverse prediction targets and how to combine them if they are complementary. We sought to answer this question by designing and evaluating a new DL ensembling method called MULTICOM_LIGAND for protein-ligand modeling, which we will describe in detail in Section 7.3.

Our MULTICOM_LIGAND team (group number 207) participating in the 2024 CASP16 experiment submitted 33 models for 13 incidental ligand pose targets; 1,165 models for 233 ligand pose pharma targets; 700 models for 140 ligand affinity pharma targets; and 110 affinity predictions for 110 phase-2 ligand affinity pharma targets, representing a submission for every protein target available in the CASP16 ligand prediction category. To facilitate such a breadth of prediction types, we designed MULTICOM_LIGAND as a modular software framework for protein-ligand modeling. Originally developed as a DL benchmarking toolkit for protein-ligand docking methods (i.e., PoseBench [172]), we adapted the core predictor modules of this benchmarking pipeline to support the prediction of arbitrary protein-ligand structures with associated confidence and affinity scores from protein sequence and ligand SMILES string inputs.

Moreover, to provide high-accuracy estimates of a protein-ligand complex's binding affinity from only primary sequences, we concurrently developed the new FLOWDOCK generative flow matching model for joint prediction of protein-ligand structure and binding affinity [192]. Notably, our initial development of FLOWDOCK revealed that joint training and prediction of protein-ligand structure and binding affinity yielded top results in various internal affinity prediction benchmarks we used for model prototyping and evaluation. As such, in the CASP16 experiment, we integrated FLOWDOCK into MULTICOM_LIGAND as an optional add-on of the framework that, as one desires, can use initially provided protein-ligand complex structures as additional model inputs for confidence and affinity estimation. This flexible soft-

ware design greatly simplified our usage of MULTICOM_LIGAND for Stage 2 of the CASP16 binding affinity prediction category, in which predictors were given the crystal structure of a protein-ligand complex and asked to estimate the complex's binding affinity using this additional information.

According to the CASP16 experiment's official analysis, in the protein-ligand structure prediction category, MULTICOM_LIGAND ranked fifth with its predictions' median lDDT-PLI score of 0.58, which denotes a protein-ligand interaction (PLI)-focused implementation of the local Distance Difference Test (lDDT) for assessment of biomolecular structure accuracy. Further, in the protein-ligand binding affinity prediction category, MULTICOM_LIGAND achieved a Kendall's Tau ranking coefficient of 0.32 in Affinity Stage 1, earning it fifth place overall. Notably, MULTICOM_LIGAND performed better than many CASP16 template-based predictors, demonstrating that deep learning has advanced the state of the art of protein-ligand structure and binding affinity prediction since CASP15.

## 7.3 METHODS

### 7.3.1 Overview of approach

From primary sequence inputs of a protein and one or more ligands alone, MULTICOM_LIGAND, visualized in Figure 7.1, provides users with rank-ordered predicted protein-ligand complex conformations filtered using structural and chemical sanity checks available in the PoseBusters software suite [112] and annotated with estimated per-atom quality scores and binding affinity values produced by our new generative flow matching model FLOWDOCK. This approach is generally summarized in Algorithm 2. The steps of the approach are described in detail in the following subsections.

**Algorithm 2** MULTICOM_LIGAND for protein-ligand structure and affinity prediction

**Notation:** ($X$: intermediate protein or protein-ligand structure; $\hat{X}$: final protein-ligand structure; $\hat{B}$: binding affinity, $\hat{C}$: confidence score)

**Input:** Protein sequence and ligand SMILES string $(S, M)$
Predict $X^{init} \leftarrow$ ESMFold$(S)$
Sample $X^{dd} \leftarrow$ DiffDock-L$(S, M, X^{init})$
Sample $X^{db} \leftarrow$ DynamicBind$(S, M, X^{init})$
Sample $X^{np} \leftarrow$ NeuralPLexer$(S, M, X^{init})$
Predict $X^{rfaa} \leftarrow$ RoseTTAFold-All-Atom$(S, M)$
Rank $X^{con} \leftarrow$ StructureConsensus$(X^{dd,db,np,rfaa})$
Bust $X^{bust} \leftarrow$ PoseBustersFilters$(X^{con})$
**if** Is Multi-Ligand **then**
    Clash Bust $X^{bust} \leftarrow$ ClashFilters$(X^{bust})$
Finalize $\hat{X}, \hat{C}, \hat{B} \leftarrow$ FlowDockAssess$(S, M, X^{bust})$
**Output:** Sampled top-5 heavy-atom structures $\hat{X}$ with confidence scores $\hat{C}$ and binding affinities $\hat{B}$

### 7.3.2 Protein-ligand inputs

MULTICOM_LIGAND represents a protein-ligand complex as a pair of single-/multi-chain protein sequence and SMILES string of one or more ligands $(S, M)$. Multiple chains within a protein sequence are delimited using the character ":", whereas multi-ligand SMILES sequences within the same string are separated using the character "." following RDKit's conventions for parsing "fragment" ligands of a single molecule [117]. Certain protein-ligand structure prediction methods employed in MULTICOM_LIGAND support using predicted protein structures as input to enhance their prediction accuracy. Accordingly, we use ESMFold [25] to provide predicted protein structure inputs to these methods. Note that, during the CASP16 experiment, we instead predicted these protein structures using AlphaFold 3 [27], though the public release of MULTICOM_LIGAND's source code by default uses the MIT-licensed ESMFold model for these purposes.

### 7.3.3 Structure prediction methods

Based on the results of our previous benchmark of DL-based protein-ligand docking methods [172], MULTICOM_LIGAND employed four representative DL methods to predict the structure for a protein-ligand sequence input: DiffDock-L [170], DynamicBind [28], RoseTTAFold-All-Atom [30], and NeuralPLexer [29]. We then grouped these methods into one of two groups, DL docking methods (i.e., DiffDock-L and DynamicBind) and DL co-folding methods (i.e., RoseTTAFold-All-Atom and NeuralPLexer), where the former group uses a predicted protein structure to perform DL-based molecular docking and the latter group predicts full protein-ligand complex conformations from primary sequence inputs. Lastly, we further subdivided these DL docking and DL co-folding groups into protein-fixed/protein-flexible categories (i.e., DiffDock-L/DynamicBind) and predictive/generative categories (i.e., RoseTTAFold-All-Atom/NeuralPLexer), respectively.

### 7.3.4 Ranking heuristics

One of the primary hypotheses driving this work is that geometrically similar ligand poses predicted by different DL methods should largely coincide with an accurate protein-ligand binding pocket and pose prediction overall. That is, when all DL methods have predicted nearly the same binding pocket and ligand pose for a given ligand molecule, they have, in essence, reached a "structural consensus" on the location and orientation of the crystal ligand pose. Based on this consensus (n.b., which may be misled if the majority of methods predict a similar incorrect binding pocket), we formulate an unsupervised ranking metric that calculates the pairwise root mean square deviation (RMSD) of all ligand poses predicted by each DL method and rank-orders the poses according to their average pairwise RMSD to each other. This provides a simple, computationally efficient heuristic (similar to that of [214] for protein complex structure ranking) for selecting our "best guess" of the location

and orientation of a ligand pose given a pool of predictions produced by various DL prediction methods.

### 7.3.5 Ligand pose filters

An important component of MULTICOM_LIGAND's design is that it not only curates a list of rank-ordered protein-ligand complex conformations produced by some of the latest DL prediction methods but also re-ranks (i.e., down-weights) its top-5 predicted conformations if any prediction fails to pass each of the standardized structural and chemical validity tests available in the PoseBusters software suite [112]. This provides an additional layer of filtering to ensure that MULTICOM_LIGAND's top predictions are ordered according to a secondary heuristic that posits that accurate ligand poses must not only be identified through a consensus of different prediction methods but must also not contain any violations of known ligand biochemistry such as non-planar ring conformations or steric clashes with protein heavy atoms.

### 7.3.6 Selected poses

During MULTICOM_LIGAND's initial stage of development, we discovered the need to add another layer of pose ranking: the possibility of encountering multi-ligand prediction targets for which accurate poses can be identified but may contain undesirable (and unrealistic) inter-ligand steric clashes between ligand heavy atoms. Notably, this phenomenon frequently occurs with DL methods such as DiffDock-L and DynamicBind which were originally trained on only *single*-ligand protein complexes, necessitating a stopgap measure to prevent such (clashing) poses from being selected as MULTICOM_LIGAND's top-ranked pose. Consequently, for multi-ligand prediction targets, MULTICOM_LIGAND automatically assigns predictions made by the DL method NeuralPLexer (n.b., which was trained on multi-ligand protein complexes with inter-ligand steric clash penalties) a higher rank than any other method's predic-

tions, to discourage (potentially) clashing poses produced by the other (single-ligand) DL methods from being selected as MULTICOM_LIGAND's top pose prediction.

### 7.3.7 Confidence & affinity prediction

A final component of MULTICOM_LIGAND's design is its ability to annotate its top-5 predicted protein-ligand structure conformations with estimated per-atom confidence scores and per-ligand binding affinity values. This is made possible by our new FLOWDOCK generative model, a version of NeuralPLexer fine-tuned with geometric flow matching for joint protein-ligand structure and binding affinity prediction. Notably, the original NeuralPLexer model was trained as a denoising diffusion probabilistic model [147, 215, 157, 212] that predicts protein(-multi)-ligand complex structures and their confidence scores from primary sequence inputs, whereas FLOWDOCK generalizes NeuralPLexer's diffusion generation framework with the emerging generative modeling framework of conditional flow matching [216, 155, 159] to enable generative (multi-ligand) structure predictions starting from biophysics-informed and empirical prior distributions [162, 167, 193].

At a high level, flow matching (n.b., as a generalization of denoising diffusion) has a DL model learn to solve an ordinary differential equation (ODE) that transforms data points derived from an easy-to-sample prior distribution $X_0$ (e.g., a Gaussian distribution) to another *empirical* distribution $X_1$ (e.g., the distribution of crystal structures in the RCSB Protein Data Bank (PDB) [173]). A DL model learns a solution to such an ODE by repeatedly "denoising" an interpolative noising schedule whereby, for a random time step $t \in [0, 1]$ sampled during training, an input data point $x_1 \in X_1$ (e.g., a 3D biomolecular crystal structure) is "noised" according to time step $t$ typically using simple linear interpolation such as $x_t = (1 - t) \cdot x_0 + t \cdot x_1$, and the model is then tasked with predicting the original version of this data point $x_1$. Once trained e.g., in the context of structure prediction, such a DL model can

be run iteratively to sample *multiple* 3D biomolecular structures for a primary input sequence starting from time step $t = 0$ representing a fully random point cloud $x_0 \in X_0$. Note that additional (e.g., Gaussian) noise is typically injected into these training and sampling processes to ensure the model produces more than a trivial mapping between point masses [162].

Importantly, the primary novelty of flow matching is that one's prior distribution can be arbitrarily chosen, in contrast to denoising diffusion for which typically only a Gaussian prior distribution can be used. This makes modeling of 3D biomolecules, in particular, much more flexible in that, with flow matching, one can specify a prior distribution informed by known biophysical properties such as a harmonic prior [162] or one derived from the outputs of another DL structure prediction model such as ESMFold [25]. As such, for a given protein sequence and ligand SMILES string, FlowDock takes precise advantage of this modeling flexibility by sampling an initial protein structure using ESMFold and an initial molecule-like ligand conformation from a harmonic prior distribution at the start of its structure prediction sampling processes, which considerably reduces its training and prediction dynamics for arbitrary protein-ligand complexes.

In addition to introducing bespoke prior distributions for structure sampling, FlowDock repurposes NeuralPLexer's frozen (i.e., non-trainable) confidence estimation module as an additional (trainable) binding affinity prediction module, which was then fine-tuned for binding affinity estimation using the well-known PDBBind 2020 dataset [174, 108]. Overall, FlowDock's model design provides a simple add-on module within MULTICOM_ligand to report (when requested) confidence scores for (predicted) protein and ligand heavy atom coordinates and binding affinity values for each ligand based on their (predicted) heavy atom coordinates (n.b., Pearson's correlation between the two: -0.127).

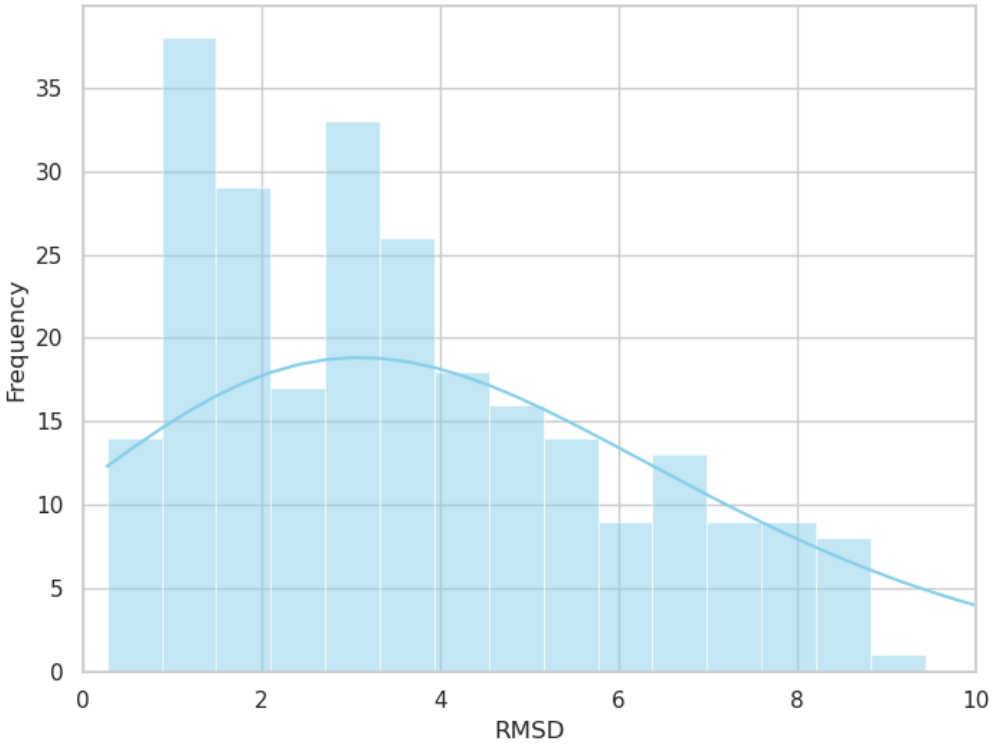
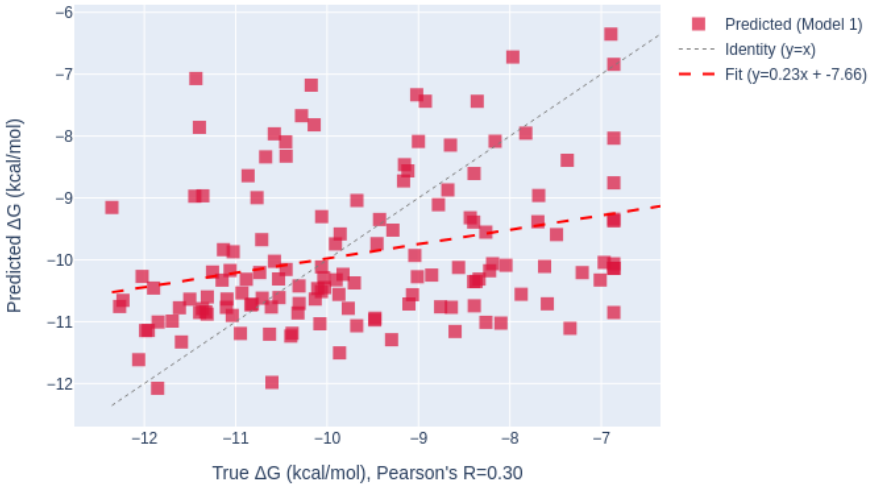


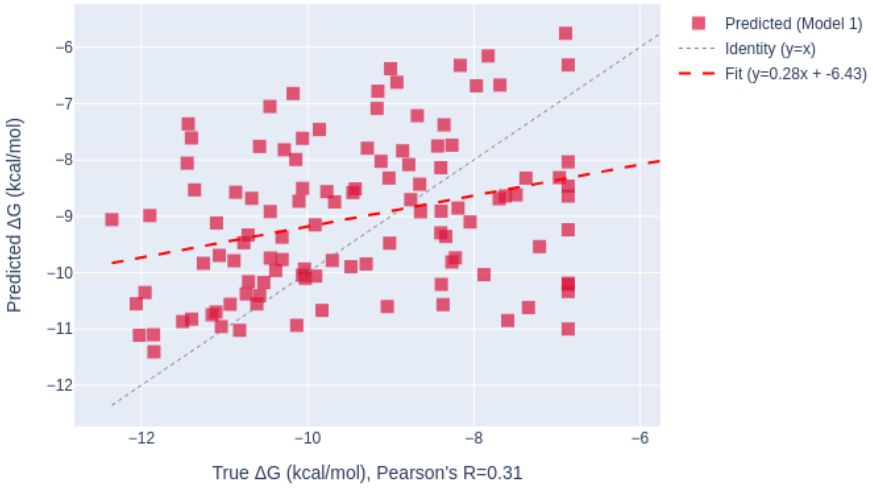

Figure 7.2: The histogram of RMSD values of MULTICOM_LIGAND's top-ranked (Model: 1) ligand models for the CASP16 protein-ligand structure prediction category (n=233). MULTICOM_LIGAND ranked 5th in this category.

## 7.4 RESULTS

The blind structure prediction benchmarking results of MULTICOM_LIGAND in the CASP16 experiment, as illustrated in Figure 7.2, demonstrate that our DL ensembling approach to protein-ligand structure modeling (n.b., ranked fifth among 34 predictor groups) reliably produces structurally accurate ligand-bound poses (∼2.5 average (Model: 1) RMSD) of the diverse, pharmaceutically relevant protein complexes available in this experiment. Furthermore, Figures 7.3a and 7.3b (for Affinity Stages 1 and 2, respectively) illustrate that MULTICOM_LIGAND's predicted protein-ligand binding affinities are modestly correlated (Pearson's R values of 0.30 and 0.31, respectively) with their ground-truth values (n.b., ranking fifth among 28 predictor groups),

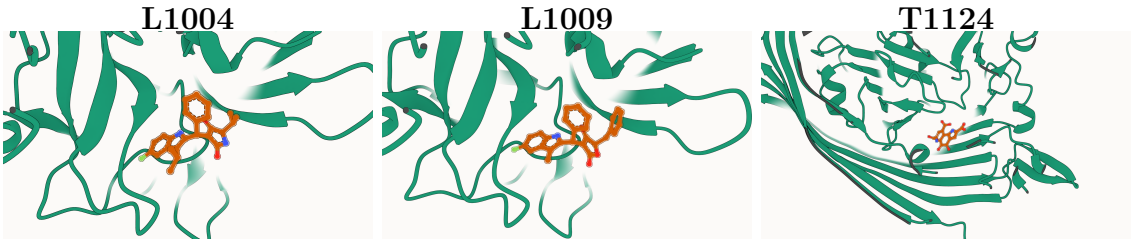



(a) MULTICOM_LIGAND's binding affinity correlation (Stage 1, n=140).




(b) MULTICOM_LIGAND's binding affinity correlation (Stage 2, n=110).

Figure 7.3: Summary of MULTICOM_LIGAND's CASP16 binding affinity prediction performance: (a) Stage 1 correlation, and (b) Stage 2 correlation. MULTICOM_LIGAND ranked fifth in this category.




(a) RMSD=0.510 (b) RMSD=0.603 (c) RMSD=4.244

Figure 7.4: MULTICOM_LIGAND's top-ranked protein-ligand complex predictions of three CASP16 ligand targets.

highlighting the real-world utility of our approach to estimating binding affinities for virtual screening in drug discovery [217].

As Figures 7.2, 7.3a, and 7.3b showcase, many of MULTICOM_LIGAND's pose and binding affinity predictions are highly accurate, yielding several predicted poses with an RMSD less than 1 and estimated affinities nearly identical to their corresponding true values. Nonetheless, in several other cases, MULTICOM_LIGAND's pose predictions yielded RMSDs above 4, indicating in these cases our approach failed to identify the correct protein-ligand binding pockets for DL-based docking. Moreover, the gaps between MULTICOM_LIGAND's predicted affinities and their true counterparts were occasionally large, suggesting in these cases MULTICOM_LIGAND was unsuccessful in differentiating weak from strong binding.

Like all other CASP16 ligand predictor groups, MULTICOM_LIGAND's affinity predictions given crystal protein-ligand structures as additional inputs in Affinity Stage 2 were not statistically significant in their differences to those of Stage 1 (Pearson's R of 0.31 vs. 0.30), highlighting that, to make accurate binding affinity predictions, FLOWDOCK's representations derived from primary sequence inputs were generally more useful to the model rather than additional structural context proved to be. In the following subsections, we examine a subset of MULTICOM_LIGAND's CASP16 pose predictions to study its relative strengths and weaknesses revealed by the experiment.

### 7.4.1 L1004

As one of the first pharma targets released to predictors for CASP16, ligand target L1004 represents a globular protein with a well-defined binding pocket for molecular docking. As such, the challenge presented by this target is largely in modeling the most accurate pose of this novel ligand *within* the pocket rather than locating the pocket itself. MULTICOM_LIGAND's top-ranked prediction for this target (Figure 7.4a) yielded a precise ligand RMSD of 0.510 and an lDDT-PLI of 0.963 (ranking 2nd overall). Interestingly, MULTICOM_LIGAND's rank-3 prediction achieved an even lower ligand RMSD of 0.483 (ranking *1st* overall), suggesting that our structural consensus ranking heuristic mislabeled our most accurate pose for this target yet still ranked it among the ensemble's top-5 predictions. The top prediction's AlphaFold 3 protein structure for L1004 had a protein backbone RMSD (BB-RMSD) of 0.224, highlighting that our DL ensemble methods each had access to a highly structurally accurate (*holo-like*) protein structure for ligand docking or pose prediction for this target, which contributed to their success in this case.

### 7.4.2 L1009

Due to the hierarchical naming structure of CASP16's pharma ligand targets, target L1009 contains the same binding pocket as target L1004 yet asks predictors to provide poses for a new and conformationally distinct ligand. MULTICOM_LIGAND's top-ranked prediction for this target (Figure 7.4b) achieved a ligand RMSD of 0.603 and an lDDT-PLI of 0.950 (ranking 2nd overall), comparable to its predictions for L1004 with high overall accuracy. Again of interest, MULTICOM_LIGAND's rank-2 prediction yielded even better results with a ligand RMSD of 0.525 (ranking *1st* overall), further emphasizing the importance in future work of identifying efficient ways of augmenting our structural consensus ranking heuristic (e.g., with FLOWDOCK's predicted confidence scores).

### 7.4.3 T1214

CASP16 incidental ligand target T1214 represents a beta barrel membrane protein structure interacting with a single PQQ ligand molecule. Figure 7.4c shows MULTICOM_LIGAND's (failed) top-ranked prediction of this target, which achieved a modest ligand heavy atom RMSD of 4.244 (n.b., 3.994 with the crystal protein structure) and lDDT-PLI of 0.357 (ranking 34th overall). As our initial AlphaFold 3 prediction of this target's beta barrel protein structure yielded a reasonable BB-RMSD of 1.687 (n.b., compared to the BB-RMSD of 0.822 achieved by the top-ranking group for this target), one possible explanation for the difficulties MULTICOM_LIGAND faced for this target is that membrane proteins constitute approximately only 5% of the PDB's composition [173]. Consequently, we posit that deep learning-based docking methods trained on common subsets of the PDB such as PDBBind [174] are likely to underperform for such targets, since their predictions are primarily optimized for docking with more common types of (e.g., helical) proteins. This suggests that MULTICOM_LIGAND's performance may be improved as new deep learning methods (in particular co-folding methods) trained on more balanced mixtures of biomolecular data are introduced.

## 7.5 DISCUSSION

In this chapter, we introduced MULTICOM_LIGAND, a deep learning-based ensembling method for protein-ligand structure prediction combined with flow matching for joint structure and binding affinity prediction. Its blind assessment results in the CASP16 experiment demonstrate its efficacy and utility for real-world drug discovery efforts. Future work could include investigating whether FLOWDOCK's predicted confidence scores could enhance the ranking performance of MULTICOM_LIGAND's structural consensus heuristic and whether the latest DL co-folding methods such as

AlphaFold 3 [27], Chai-1 [153], and NeuralPLexer 3 [194] may benefit from a DL ensembling approach like our new MULTICOM_LIGAND method or if their predictions may be augmented with additional rank-ordering and binding affinity estimations provided by lightweight generative models such as our new FLOWDOCK model [192].

# Chapter 8

# SUMMARY AND CONCLUDING REMARKS

## 8.1 CONTRIBUTIONS

This dissertation advances the modeling of 3D biomolecules with deep learning, spanning protein-protein interaction prediction, protein-binding small molecule generation, and docking pose estimation using geometric and generative methods. The algorithms, datasets, and metrics introduced herein establish a foundation for a more data-driven and learning-based approach to computational biology. Furthermore, this work has catalyzed new research directions in line graph representation learning, geometric graph neural networks, generative modeling, and the empirical analysis of geometric message passing expressivity. Collectively, these contributions have accelerated progress at the intersection of machine learning and computational biology.

## 8.2 FUTURE DIRECTIONS

Building on the findings of this dissertation, several promising research directions emerge. These include all-atom biomolecular generative modeling, improved inference-time scaling and search strategies for pre-trained biomolecular design models, and scalable reward-guided inference of flow-based generative models. Advancing these areas could have broad implications for frontier drug discovery, materials science, and energy research.

# Chapter A

# SUPPLEMENTARY MATERIALS FOR "GEOMETRIC TRANSFORMERS FOR PROTEIN INTERFACE CONTACT PREDICTION"

Adapted from Alex Morehead, Chen Chen, and Jianlin Cheng. "Geometric Transformers for Protein Interface Contact Prediction". *The Tenth International Conference on Learning Representations* (ICLR 2022).

## A.1 SAMPLE INTERFACE CONTACT PREDICTIONS

In the first row of Figure A.1, we see predictions made by DeepInteract for a homodimer complex from our test partition of DIPS-Plus (i.e., PDB ID: 4HEQ). The leftmost image represents the softmax contact probability map. The center image corresponds to the same contact map after having a 0.5 probability threshold applied to it such that residue pairs with at least a 50% probability of being in interaction with each other have their interaction probabilities rounded up to 1.0. The rightmost image is the ground-truth contact map. Similarly, in the second row of Figure A.1, we observe the cropped predictions made by DeepInteract for a CASP-CAPRI test heterodimer (i.e., PDB ID: 6TRI).

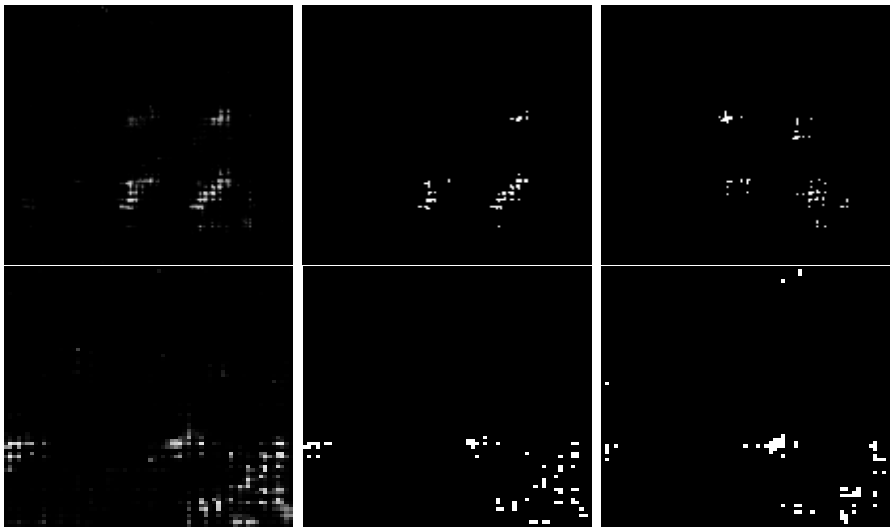


Figure A.1: DeepInteract's softmax contact probabilities (leftmost column), 0.5 positive probability-thresholded predictions (middle column), and ground-truth labels (rightmost column), respectively, for PDB ID: 4HEQ (first row) and 6TRI (second row), two of the complexes in our test datasets.

## A.2 TOP-$k$ TEST PRECISION AND RECALL OF BOTH COMPLEX TYPES IN DIPS-PLUS AND CASP-CAPRI

Formally, our definitions of a model's top-$k$ precision $prec_k$ and recall $rec_k$, where $T_{pos_k}$ represents the number of true positive residue pairs selected from a model's top-$k$ most probable pairs and $T_{pos}$ corresponds to the total number of true positive pairs in the complex, are

$$prec_k = \frac{T_{pos_k}}{k} \tag{A.1}$$

and

$$rec_k = \frac{T_{pos_k}}{T_{pos}}. \tag{A.2}$$

After defining top-$k$ recall as such, in Tables A.1 and A.2 we provide the results of each model's top-$k$ recall in the same set of experiments as given in the Results section of Chapter 2.

Table A.1: The average top-$k$ recall on two types of DIPS-Plus test targets.

| | 16 (Homo) | | | 16 (Hetero) | | |
|---|---|---|---|---|---|---|
| Method | R@$L$ | R@$L/2$ | R@$L/5$ | R@$L$ | R@$L/2$ | R@$L/5$ |
| BI | 0.01 | 0 | 0 | 0.01 | 0.01 | 0.01 |
| DH | 0.07 | 0.04 | 0.02 | | | |
| CC | | | | **0.17** | **0.12** | **0.07** |
| DI (GCN) | 0.14 (0.03) | 0.08 (0.01) | 0.04 (0.01) | 0.08 (0.02) | 0.05 (0.02) | 0.02 (0.01) |
| DI (GT) | 0.17 (0.01) | 0.10 (0.01) | 0.05 (0.01) | 0.09 (0.02) | 0.05 (0.02) | 0.03 (0.01) |
| DI (GeoT w/o EPE) | 0.18 (0.02) | 0.11 (0.01) | 0.05 (0.01) | 0.11 (0.03) | 0.07 (0.02) | 0.03 (0.02) |
| DI (GeoT w/o GFG) | 0.19 (0.04) | 0.11 (0.03) | 0.05 (0.02) | 0.09 (0.01) | 0.05 (0.02) | 0.03 (0.01) |
| DI (GeoT) | **0.19** (**0.004**) | **0.12** (**0.004**) | **0.06** (**0.003**) | 0.12 (0.003) | 0.07 (0.01) | 0.03 (0.01) |

Table A.2: The average top-$k$ recall on dimers from CASP-CAPRI 13 & 14.

| | 14 (Homo) | | | 5 (Hetero) | | |
|---|---|---|---|---|---|---|
| Method | R@$L$ | R@$L/2$ | R@$L/5$ | R@$L$ | R@$L/2$ | R@$L/5$ |
| BI | 0.02 | 0.01 | 0 | 0.01 | 0 | 0 |
| DH | 0.02 | 0.01 | 0 | | | |
| CC | | | | 0.03 | 0.01 | 0.01 |
| DI (GCN) | 0.10 (0.01) | 0.07 (0.01) | 0.04 (0.02) | 0.08 (0.04) | 0.04 (0.02) | 0.02 (0.01) |
| DI (GT) | 0.10 (0.01) | 0.06 (0.01) | 0.02 (0.01) | 0.10 (0.01) | 0.05 (0.01) | 0.02 (0.01) |
| DI (GeoT w/o EPE) | 0.11 (0.01) | 0.07 (0.01) | 0.04 (0.01) | 0.12 (0.02) | 0.07 (0.01) | 0.03 (0.01) |
| DI (GeoT w/o GFG) | 0.10 (0.02) | 0.06 (0.01) | 0.03 (0.01) | 0.11 (0.02) | 0.07 (0.01) | 0.03 (0.01) |
| DI (GeoT) | **0.12** (**0.03**) | **0.07** (**0.01**) | **0.04** (**0.01**) | **0.15** (**0.02**) | **0.09** (**0.01**) | **0.04** (**0.01**) |

## A.3 DEFINITION OF EDGE GEOMETRIC FEATURES

Similar to [53], we construct a local reference frame (i.e., an orientation $\boldsymbol{O}_i$) for each protein chain graph's residues. Representing each residue by its Cartesian coordinates $\boldsymbol{x}_i$, we formally define

$$\boldsymbol{u}_i = \frac{\boldsymbol{x}_i - \boldsymbol{x}_{i-1}}{\|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\|}, \quad \boldsymbol{n}_i = \frac{\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|}, \quad \boldsymbol{b}_i = \frac{\boldsymbol{u}_i - \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i - \boldsymbol{u}_{i+1}\|}. \tag{A.3}$$

with $\boldsymbol{n}_i$ being the unit vector normal to the plane formed by the rays $(\boldsymbol{x}_{i-1} - \boldsymbol{x}_i)$ and $(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i)$ and $\boldsymbol{b}_i$ being the negative bisector of this plane. We then define $\boldsymbol{O}_i$ as

$$\boldsymbol{O}_i = [\boldsymbol{b}_i \;\; \boldsymbol{n}_i \;\; \boldsymbol{b}_i \times \boldsymbol{n}_i]. \tag{A.4}$$

Having defined the orientation $\boldsymbol{O}_i$ for each residue that describes the local reference frame $(\boldsymbol{x}_i,\ \boldsymbol{O}_i)$. To provide the Geometric Transformer with an alternative notion of residue-residue orientations, we define the unit vector normal to the amide plane for residue $i$ as

$$\boldsymbol{U}_i = (\boldsymbol{x}_{C\alpha_i} - \boldsymbol{x}_{C\beta i}) \times (\boldsymbol{x}_{C\beta_i} - \boldsymbol{x}_{N_i}) \tag{A.5}$$

where $\boldsymbol{x}_{C\alpha_i}$, $\boldsymbol{x}_{C\beta i}$, and $\boldsymbol{x}_{Ni}$ are the Cartesian coordinates of the residue's carbon-alpha ($C\alpha$), carbon-beta ($C\beta$), and nitrogen ($N$) atoms, respectively.

Finally, we relate the reference frames for residues $i$ and $j$ by describing their edge geometric features as

$$\left(\boldsymbol{r}(\|\boldsymbol{x}_j - \boldsymbol{x}_i\|),\;\; \boldsymbol{O}_i^T \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{\|\boldsymbol{x}_j - \boldsymbol{x}_i\|},\;\; \boldsymbol{q}(\boldsymbol{O}_i^T \boldsymbol{O}_j),\;\; \boldsymbol{a}(\boldsymbol{U}_i, \boldsymbol{U}_j)\right) \tag{A.6}$$

with the first term $\boldsymbol{r}()$ being a distance encoding of 16 Gaussian radial basis functions spaced isotropically from 0 to 20 Å, the second term describing the relative direction of $\boldsymbol{x}_j$ with respect to reference frame $(\boldsymbol{x}_i, \boldsymbol{O}_i)$, the third term detailing an orientation encoding $\boldsymbol{q}()$ of the quaternion representation of the rotation matrix $\boldsymbol{O}_i^T \boldsymbol{O}_j$, representing each quaternion with respect to its vector of real coefficients, and the fourth term $\boldsymbol{a}()$ representing the angle between the amide plane normal vectors $\boldsymbol{U}_i$ and $\boldsymbol{U}_j$.

Our definition of these edge geometric features makes use of the backbone atoms for each residue. As such, the graph representation of protein chains we use with the Geometric Transformer encodes not only residue-level geometric features but also those derived from an atomic view of protein structures. We hypothesized this hybrid approach to modeling protein structure geometries would have a notice-

Table A.3: The protein complexes selected from DIPS-Plus for testing interface contact predictors.

| PDB ID | Chain 1 | Chain 2 | Type | PDB ID | Chain 1 | Chain 2 | Type |
|---|---|---|---|---|---|---|---|
| 1BHN | B | D | Homo | 1AON | R | S | Hetero |
| 1KPT | A | B | Homo | 1BE3 | D | E | Hetero |
| 1SDU | A | B | Homo | 1GK8 | K | M | Hetero |
| 1UZN | A | B | Homo | 1OCZ | R | V | Hetero |
| 2B4H | A | B | Homo | 1UWA | A | I | Hetero |
| 2G30 | C | E | Homo | 3A6N | A | E | Hetero |
| 2GLM | E | F | Homo | 3ABM | D | K | Hetero |
| 2IUO | D | J | Homo | 3JRM | H | I | Hetero |
| 3BXS | A | B | Homo | 3MG6 | D | E | Hetero |
| 3CT7 | B | E | Homo | 3MNN | C | F | Hetero |
| 3NUT | A | D | Homo | 3T1Y | E | H | Hetero |
| 3RE3 | B | C | Homo | 3TUY | D | E | Hetero |
| 4HEQ | A | B | Homo | 3VYG | G | H | Hetero |
| 4LIW | A | B | Homo | 4A3D | C | L | Hetero |
| 4OTA | D | F | Homo | 4CW7 | G | H | Hetero |
| 4TO9 | B | D | Homo | 4DR5 | G | I | Hetero |

able downstream effect on interface contact prediction precision via the node and edge representations learned by the Geometric Transformer. This hypothesis is confirmed in the Results section of Chapter 2.

## A.4 PROTEIN COMPLEXES SELECTED FOR TESTING

To facilitate reproducibility of the results presented in the Results section of Chapter 2, Table A.3 displays the PDB and chain IDs of DIPS-Plus protein complexes chosen for testing. Likewise, in Table A.4, we provide the PDB and chain IDs of CASP-CAPRI 13-14 targets chosen for testing. These two tables describe precisely which targets were selected and ultimately used in our RCSB-derived benchmarks. For full data provenance, the targets we selected from the Docking Benchmark 5 dataset [41]

Table A.4: The CASP-CAPRI 13-14 protein complexes selected for testing interface contact predictors.

| PDB ID | Chain 1 | Chain 2 | Type |
|---|---|---|---|
| 5W6L | A | B | Homo |
| 6D2V | A | B | Homo |
| 6E4B | A | B | Homo |
| 6FXA | C | D | Homo |
| 6HRH | A | B | Homo |
| 6MXV | A | B | Homo |
| 6N64 | A | B | Homo |
| 6N91 | A | B | Homo |
| 6NQ1 | A | B | Homo |
| 6QEK | A | B | Homo |
| 6UBL | A | B | Homo |
| 6UK5 | A | B | Homo |
| 6YA2 | A | B | Homo |
| 7CWP | C | D | Homo |
| 6CP8 | A | C | Hetero |
| 6D7Y | A | B | Hetero |
| 6TRI | A | B | Hetero |
| 6XOD | A | B | Hetero |
| 7M5F | A | C | Hetero |

for benchmarking are the same 55 protein heterodimers used for testing in works such as those of [40], [42], and [35].

## A.5 INVARIANCE OR EQUIVARIANCE?

In our view, a natural question to ask concerning a deep learning architecture designed for a specific task is whether equivariance to translations and rotations in $\mathbb{R}^3$ should be preferred over invariance to transformations in such a geometric space. The benefits of employing equivariant representations in a deep learning architecture primarily include symmetry-preserving updates to type-1 tensors such as the coor-

dinates representing an object in $\mathbb{R}^3$ and the derivation of invariant relative feature poses for type-0 features such as scalars [218]. However, equivariant representations, particularly those derived with a self-attention mechanism, can induce large memory requirements for training and inference. In contrast, in the context of data domains such as ordered sets or proteins where there exists a canonical ordering of points, invariant representations may be adopted to simultaneously reduce memory requirements and provide many of the benefits of using equivariant representations such as attaining these relative poses of type-0 features [53, 15]. As such, in the context of the Geometric Transformer, we opted to pursue invariance over equivariance, to reduce the network's effective memory requirements while improving its learning efficiency and generalization capabilities [26]. However, for applications such as protein-protein docking that may more directly rely on type-1 tensors for network predictions [47], designing one's network architecture to preserve full translation and rotation equivariance in $\mathbb{R}^3$ is, in our perspective, a worthwhile research direction to pursue as many promising results on molecular datasets have already been demonstrated with equivariant neural networks such as SE(3)-Transformers [219] and lightweight graph architectures such as the Equivariant Graph Neural Network [18].

## A.6 RATIONALE BEHIND THE NODE INITIALIZATION SCHEME

DIPS-Plus residue-level features are initially embedded in our protein chain graphs to accelerate the network's training. However, we also initially append node-wise min-max positional encodings in our network's operations. We do this to initialize the Geometric Transformer with information concerning the residue ordering of the chain's underlying sequence, as such ordering is important to understanding downstream protein structural, interactional, and functional properties of each residue.

## A.7 RATIONALE BEHIND THE EDGE INITIALIZATION MODULE'S DESIGN

For the edge initializer module's four protein geometric features, we sought to include enough geometric information for the network to be able to uniquely determine the Euclidean positions of each node's neighboring nodes. For this reason, we adopt similar distance, direction, and orientation descriptors as [53]. We concatenate the protein backbone-geometric features provided by inter-residue distances, directions, and orientations with the angles between each residue pair's amide plane normal vectors. This is done ultimately to apply gating to each edge's messages, distances, directions, orientations, and amide angles separately to encourage the network to learn the importance of specific channels in each of these input features. Gating is a technique that has previously been shown to encourage neural networks to not become over-reliant on any particular input feature [220] and, as such, in the GEOMETRIC TRANSFORMER can be seen as a form of channel-wise dropout for single feature sets. By also employing residual connections from original edge representations to gating-learned edge representations, the network module can operate more stably in the presence of multiple neural network layers [221]. Furthermore, in the edge initialization module, we introduce edge-wise sinusoidal position encodings to provide the network with a directional notion of residue-to-residue distances in protein chains' underlying sequences.

## A.8 RATIONALE BEHIND THE CONFORMATION MODULE'S DESIGN

The conformation module's design was inspired, in part, by SphereNet [222] and similar graph neural network architectures designed for learning on 3D graphs. What distinguishes our conformation module from the works of others is its introduction of

the notion of $2n$ edge geometric neighborhoods when updating edge representations as well as its incorporation of geometric insights specific to large biomolecules such as proteins. Namely, by including the residue-residue distances, residue-residue local reference frame directions and (quaternion) orientations, and amide plane-amide plane angles, the network is provided with enough information to ascertain the relative coordinates of each neighboring residue from a given residue's local reference frame [222], thereby ensuring the network's capability of adequately learning from 3D structures.

## A.9 ALTERNATIVE NETWORKS WITHIN THE INTERACTION MODULE

We, like [35], note that the task of interface prediction bears striking similarities to dense prediction tasks in computer vision (e.g., semantic segmentation). In this train of thought, we experimented with several semantic segmentation models as replacements for our interaction module's dilated ResNet, one namely being DeepLabV3Plus [223]. We observed a strong propensity of such semantic segmentation models to identify interaction regions well but to do so with low *pixel-wise* precision. We hypothesize this is due to the downsampling and upsampling methods often employed within such architectures that invariably degrade the original input tensor's representation resolution. We also experimented with several state-of-the-art Vision Transformer and MLP-based models for computer vision but ultimately found their algorithmic complexity, memory usage, or input shape requirements to be prohibitive for this task, since our test datasets' input protein complexes can vary greatly in size to contain between 20 residues and over 2,000 residues in length. As such, for the design of DeepInteract's interaction module, we experimented primarily with convolution-based architectures that do not employ such sampling techniques or pose limited input size constraints.

## A.10 HARDWARE USED

The Oak Ridge Leadership Facility (OLCF) at the Oak Ridge National Laboratory (ORNL) is an open science computing facility that supports HPC research. The OLCF houses the Summit compute cluster. Summit, launched in 2018, delivers 8 times the computational performance of Titan's 18,688 nodes, using only 4,608 nodes. Like Titan, Summit has a hybrid architecture, and each node contains multiple IBM POWER9 CPUs and NVIDIA Volta GPUs all connected with NVIDIA's high-speed NVLink. Each node has over half a terabyte of coherent memory (high bandwidth memory + DDR4) addressable by all CPUs and GPUs plus 800GB of non-volatile RAM that can be used as a burst buffer or as extended memory. To provide a high rate of I/O throughput, the nodes are connected in a non-blocking fat-tree using a dual-rail Mellanox EDR InfiniBand interconnect. We used the Summit compute cluster to train all our models.

## A.11 SOFTWARE USED

In addition, we used Python 3.8 [224], PyTorch 1.7.1 [225], and PyTorch Lightning 1.4.8 [226] to run our deep learning experiments. PyTorch Lightning was used to facilitate model checkpointing, metrics reporting, and distributed data parallelism across 72 Tesla V100 GPUs. A more in-depth description of the software environment used to train and predict with DeepInteract models can be found on GitHub at `https://github.com/BioinfoMachineLearning/DeepInteract`.

# Chapter B

# SUPPLEMENTARY MATERIALS FOR "GEOMETRY-COMPLETE PERCEPTRON NETWORKS FOR 3D MOLECULAR GRAPHS"

Adapted from Alex Morehead and Jianlin Cheng. "Geometry-complete perceptron networks for 3D molecular graphs". *Bioinformatics* 40.2 (2024): btae087.

## B.1 EXPANDED METHODOLOGY DISCUSSION

As a continuation of our methodological overview of GCPNet given in the main text in Section 3.3.1, here we further describe the equivariance, geometric self-consistency, and geometric completeness constraints that GCPNet satisfies.

As discussed in Section 3.3.1 of the main text, our GCPNet function $\mathbf{\Phi}$ guarantees, by design, *SE(3) equivariance* with respect to its vector-valued input coordinates and features (i.e., $x_i \in \mathbf{X}$, $\chi_i \in \boldsymbol{\chi}$, and $\xi_{ij} \in \boldsymbol{\xi}$) and *SE(3)-invariance* regarding its scalar features (i.e., $h_i \in \mathbf{H}$ and $e_{ij} \in \mathbf{E}$). In addition, $\mathbf{\Phi}$'s scalar graph representations achieve *geometric self-consistency* for the 3D structure of the input molecular graph $\mathcal{G}$, sensitizing them to the effects of molecular chirality while making them uniquely identifiable under 3D rotations. Lastly, geometric completeness requires methods that accept 3D molecular graph inputs to be able to discern the local geometric environment of a given atom with no directional ambiguities. This enables geometry-complete methods such as $\mathbf{\Phi}$ to detect the presence and influence of global

force fields acting on the graph inputs. Note that we more carefully formalize these equivariance, geometric self-consistency, and geometric completeness constraints using three corresponding definitions in Section 3.3.1 of the main text.

### B.1.1 SE(3)-equivariant complete representations

As described in the three definitions referenced above, representation learning on 3D molecular structures is a challenging task for a variety of reasons: (1) an expressive representation learning model should be able to predict arbitrary vector-valued quantities for each atom and atom pair in the molecular structure (e.g., using $\boldsymbol{\chi}'$ and $\boldsymbol{\xi}'$ to predict side-chain atom positions and atom-atom displacements for each residue in a 3D protein graph); (2) arbitrary rotations or translations to a 3D molecular structure should affect only the vector-valued representations a model assigns to a molecular graph's nodes or edges, whereas such 3D transformations of the molecular structure should not affect the model's scalar representations for nodes and edges [77]; (3) the geometrically invariant properties of a molecule's 3D structure should be uniquely identifiable by a model; and (4) in a geometry-complete manner, scalar and vector-valued representations should mutually exchange information between nodes and edges during a model's forward pass for a 3D input graph, as these information types can be correlatively related (e.g., a scalar feature such as the $L_2$ norm of a vector $v$ can be associated with the vector of origin $v$) [86, 72].

In line with this reasoning, we need to ensure that the coordinates our model predicts for the node positions in a molecular graph $\mathcal{G}$ transform according to SE(3) transformations of the input positions. This runs in contrast to previous methods that remain strictly E(3)-equivariant or E(3)-invariant to 3D transformations of the input $\mathcal{G}$ and consequently ignore the important effects of molecular chirality. At the same time, the model should jointly update the scalar and vector-valued features of $\mathcal{G}$ according to their respective molecular symmetry groups to increase the model's ex-

pressiveness in approximating geometric and physical quantities [227]. To increase its generalization capabilities, the model should also disambiguate any *geometric directions* within its local node environments and should maintain SE(3)-invariance of its scalar representations when the input graph is transformed in 3D space. Following [79], this helps prevent the model from losing important geometric or chiral information (i.e., becoming geometrically self-inconsistent) during graph message-passing. One way to do this is to introduce a new type of message-passing neural network such as GCPNet, as we have proposed in this work.

### B.1.2 Geometry-complete graph convolution with GCPNet

As a continuation of Section 3.3.3 in the main text in which we will now give a more detailed derivation of how one can perform 3D graph convolution using GCPNet, let $\mathcal{N}$(i) denote the neighbors of node $n_i$, selected using a distance-based metric such as k-nearest neighbors or a radial distance cutoff. Subsequently, we define a single layer $l$ of geometry-complete graph convolution as

$$n_i^l = \phi^l(n_i^{l-1}, \mathcal{A}_{\forall j \in \mathcal{N}(i)} \Omega_\omega^l(n_i^{l-1}, n_j^{l-1}, e_{ij}, \mathcal{F}_{ij})), \tag{B.1}$$

where $n_i^l = (h_i^l, \chi_i^l)$; $e_{ij} = (e_{ij}^0, \xi_{ij}^0)$; $\mathbf{\Phi}$ is a trainable function denoted as **GCPConv**; $l$ signifies the representation depth of the network; $\mathcal{A}$ is a permutation-invariant aggregation function; and $\Omega_\omega$ represents a message-passing function corresponding to the $\omega$-th **GCP** message-passing layer. We proceed to expand on the operations of each graph convolution layer as follows.

To start, messages between source nodes $i$ and neighboring nodes $j$ are first constructed as

$$m_{ij}^0 = \mathbf{GCP}(n_i^0 \cup n_j^0 \cup e_{ij}, \mathcal{F}_{ij}) \tag{B.2}$$

where $\cup$ denotes a concatenation operation. Then, up to the $\omega$-th iteration, each

message is updated by the $m$-th message update layer using residual connections as

$$\Omega_{\omega}^{l} = \mathbf{ResGCP}_{\omega}^{l}(m_{ij}^{l-1}, \mathcal{F}_{ij}), \tag{B.3}$$

$$\mathbf{ResGCP}_{\eta}^{l}(z_{i}^{l-1}, \mathcal{F}_{ij}) = z_{i}^{l-1} + \mathbf{GCP}_{\eta}^{l}(z_{i}^{l-1}, \mathcal{F}_{ij}), \tag{B.4}$$

where we empirically find such residual connections between message representations to reduce oversmoothing within GCPNet by mitigating the problem of vanishing gradients.

Updated node features $\hat{n}^{l}$ are then derived residually using an aggregation of generated messages as

$$\hat{n}^{l} = n^{l-1} + f(\{\Omega_{\omega,v_i}^{l} | v_i \in \mathcal{V}\}), \tag{B.5}$$

where $f$ represents an aggregation function such as a summation or mean that is invariant to permutations of node ordering. The residual connection between $\hat{n}^{l}$ and $n^{l}$ is established here to encourage the network to update the representation space of node features in a layer-asynchronous manner.

To encourage GCPNet to make its node feature representations independent of the size of each input graph, we then employ a node-centric feed-forward network to update node representations. Specifically, we apply to $\hat{n}^{l}$ a linear **GCP** function with shared weights $\phi_f$ followed by $r$ **ResGCP** modules, operations concisely portrayed as

$$\tilde{n}_{r-1}^{l} = \phi_{f}^{l}(\hat{n}^{l}) \tag{B.6}$$

$$n^{l} = \mathbf{ResGCP}_{r}^{l}(\tilde{n}_{r-1}^{l}). \tag{B.7}$$

Lastly, if one desires to update the positions of each node in $\mathcal{G}$ (e.g., as we do for tasks involving position-related predictions such as NMS), we propose a flexible, SE(3)-

equivariant method to do so using a dedicated **GCP** module as follows:

$$(h_{p_i}^l, \chi_{p_i}^l) = \mathbf{GCP}_p^l(n_i^l, \mathcal{F}_{ij}) \tag{B.8}$$

$$x_i^l = x_i^{l-1} + \chi_{p_i}^l, \text{where } \chi_{p_i}^l \in \mathbb{R}^{1\times 3}. \tag{B.9}$$

## B.2 PROOFS

### B.2.1 Proof of Proposition 1

*Proof.* Suppose the vector-valued features given to the corresponding **GCPConv** layers in GCPNet are node features $\chi_i$ and edge features $\xi_{ij}$ that are O(3)-equivariant (i.e., 3D rotation and reflection-equivariant) by way of their construction. Additionally, suppose the scalar-valued features given to the respective **GCPConv** layers in GCPNet are E(3)-invariant (i.e., 3D rotation, reflection, and translation-invariant) node features $h_i$ and edge features $e_{ij}$.

**Translation equivariance.** In line with [77], the **Centralize** operation on Line 2 of Algorithm 1 in the main text first ensures that $\mathbf{X}^0$ becomes 3D translation invariant by the following procedure. Let $\mathbf{X}(t) = (\mathbf{x}_1(t), ..., \mathbf{x}_n(t))$ represent a many-body system at time $t$, where the centroid of the system is defined as

$$C(t) = \frac{\mathbf{x}_1(t) + ... + \mathbf{x}_n(t)}{n}. \tag{B.10}$$

Note that in uniformly translating the position of the system by a vector $\mathbf{v}$, we have $\mathbf{X}(t) + \mathbf{v} \longrightarrow C(t) + \mathbf{v}$, meaning that the centroid of the system translates in the same manner as the system itself. However, note that if at time $t = 0$ we recenter the origin of $\mathbf{X}$ to its centroid, we have

$$\mathbf{X}(t) - C(0) \xrightarrow{\text{translation by } \mathbf{v} \text{ at } t=0} \mathbf{X}(t) - C(0)$$

which implies the system $\mathbf{X}$ is translation-invariant under the centralized reference $\mathbf{X}(t) - C(0)$ when the translation vector $\mathbf{v}$ is applied to $\mathbf{X}$ at time $t = 0$. Concretely, in the case of translation-invariant tasks such as predicting molecular properties or classifying point clouds, here we have successfully achieved 3D translation invariance. Moreover, for translation-equivariant tasks such as forecasting the positions of a many-body system, we can achieve translation equivariance by simply adding $C(0)$ back to the predicted positions. Therefore, using the above methodology, GCPNets are translation equivariant.

**Permutation equivariance.** Succinctly, we note that since GCPNet operates on graph-structured input data, permutation equivariance is guaranteed by design. For further discussion of why our proposed method as well as why other graph-based algorithms proposed previously are inherently permutation-equivariant, we refer readers to [228]. Therefore, GCPNets are permutation-equivariant.

**SO(3)-equivariant frames.** On Line 3 of Algorithm 1 in the main text, the **Localize** operation constructs SO(3)-equivariant (i.e., 3D rotation-equivariant) frames $\boldsymbol{\mathcal{F}}_{ij}$ in the following manner.

Define our frame encodings as

$$\boldsymbol{\mathcal{F}}_{ij}^{t} = (a_{ij}^{t}, b_{ij}^{t}, c_{ij}^{t}), \tag{B.11}$$

where we have

$$a_{ij}^{t} = \frac{x_i^t - x_j^t}{\|x_i^t - x_j^t\|}, b_{ij}^{t} = \frac{x_i^t \times x_j^t}{\|x_i^t \times x_j^t\|}, c_{ij}^{t} = a_{ij}^{t} \times b_{ij}^{t}. \tag{B.12}$$

The proof that $\boldsymbol{\mathcal{F}}_{ij}^{t}$ is equivariant under SO(3) transformations of its input space is included in [77]. However, for completeness, we include a version of it here.

Let $g \in SO(3)$ be an action under which the positions in $X$ transform equivari-

antly, and $\boldsymbol{\mathcal{F}}_{ij}^t$ be defined as we have it in Equation B.11 above. That is, we have

$$(\mathbf{x}_1(t), ..., \mathbf{x}_n(t)) \xrightarrow{g} (g\mathbf{x}_1(t), ..., g\mathbf{x}_n(t)),$$

where from the definition of $a_{ij}^t$ in Equation B.12 we have

$$a_{ij}^t \xrightarrow{g} g a_{ij}^t.$$

Considering $b_{ij}^t$, from Equation B.12 we have

$$\begin{aligned}(gx_i(t)) \times (gx_j(t)) &= \det(g)(g^T)^{-1}(x_i(t) \times x_j(t)) \\ &= g(x_i(t) \times x_j(t)),\end{aligned} \tag{B.13}$$

where using $g^{-1} = g^T$ for the orthogonal matrix $g$ gives us Equation B.13. Consequently, $b_{ij}^t \xrightarrow{g} g b_{ij}^t$. Lastly, by applying Equation B.13 once again, we have that $c_{ij}^t \xrightarrow{g} g c_{ij}^t$.

Moreover, note that under reflections of $x$, we have $R : x \to -x$ which gives us $a_{ij}^t \to -a_{ij}^t$. Thereafter, by the right-hand rule, the cross product of two equivariant vectors gives us a pseudo-vector $b_{ij}^t = x_i^t \times x_j^t \to b_{ij}^t$, where subsequently it is implied that $c_{ij}^t \to -c_{ij}^t$. Consequently, we have $\det(-a_{ij}^t, b_{ij}^t, -c_{ij}^t) = 1$, informing us that the frame encodings $\boldsymbol{\mathcal{F}}_{ij}^t$ are rotation-equivariant yet <u>not</u> reflection-equivariant (a symmetry that is important to not enforce when learning representations of chiral molecules such as proteins). Therefore, the frame encodings within GCPNet are SO(3)-equivariant.

Note, after the construction of these frames, that they are used on Line 4 of Algorithm 1 in the main text to embed all node and edge features (i.e., $h_i$, $e_{ij}$, $\chi_i$, and $\xi_{ij}$) using a single **GCP** module as well as in all subsequent **GCP** modules. We will now prove that the feature updates each **GCP** module makes with the frame

encodings $\boldsymbol{\mathcal{F}}_{ij}^t$ defined in Equation B.11 are SO(3)-equivariant.

**SO(3)-equivariant GCP module.** The operations of a **GCP** module are illustrated in Figure 3.2 in the main text and derived in Section 3.3.2 in the main text. Their SO(3) invariance for scalar feature updates and SO(3) equivariance for vector-valued feature updates is proven as follows.

Following the proof of O(3) equivariance for the GVP module in [17], the proof of SO(3) equivariance within the **GCP** module is similar, with the following modifications. Within the **GCP** module, the vector-valued features (processed separately for nodes and edges) are fed not only through a bottleneck block comprised of downward and upward projection matrices $\mathbf{D}_{\mathbf{z}}$ and $\mathbf{U}_{\mathbf{z}}$ but are also fed into a dedicated downward projection matrix $\mathbf{D}_{\mathcal{S}}$. The output of matrix multiplication between O(3)-equivariant vector features and $\mathbf{D}_{\mathcal{S}}$ yields O(3)-equivariant vector features $v_{i_{\mathcal{S}}}$ that are used as unique inputs for an SO(3)-invariant scalarization operation. In particular, the following demonstrates the invariance of our design for matrix multiplication with our **GCP** module's projection matrices (e.g., $\mathbf{D}_{\mathcal{S}}$). Suppose $\mathbf{W}_h \in \mathbb{R}^{h \times v}$, $\mathbf{V} \in \mathbb{R}^{v \times 3}$, and $\mathbf{Q} \in SO(3) \in \mathbb{R}^{3 \times 3}$. In line with [17], observe for $\mathbf{D} = (\mathbf{Q}\mathbf{V}^{\mathrm{T}}) \in \mathbb{R}^{3 \times v}$ that

$$\|\mathbf{W}_h \mathbf{D}^{\mathrm{T}}\|_2 = \|\mathbf{W}_h (\mathbf{V}^{\mathrm{T}})^{\mathrm{T}}\|_2 = \|\mathbf{W}_h \mathbf{V}\|_2.$$

Specifically, our SO(3)-invariant scalarization operation is defined as

$$q_{ij} = (v_{i_{\mathcal{S}}} \cdot a_{ij}^t, v_{i_{\mathcal{S}}} \cdot b_{ij}^t, v_{i_{\mathcal{S}}} \cdot c_{ij}^t), \tag{B.14}$$

where $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t)$ denotes the SO(3)-equivariant frame encodings defined in Equations B.11 and B.12.

To prove that Equation B.14 yields SO(3)-invariant scalar features, let $g \in SO(3)$ be an arbitrary orthogonal transformation. Then we have $v_{i_{\mathcal{S}}} \to g v_{i_{\mathcal{S}}}$, and similarly $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t) \to (g a_{ij}^t, g b_{ij}^t, g c_{ij}^t)$. Now, similar to [77], we can derive that

Equation B.14 becomes

$$
\begin{aligned}
&(v_{i_S} \cdot a_{ij}^t, v_{i_S} \cdot b_{ij}^t, v_{i_S} \cdot c_{ij}^t) \\
&\quad \rightarrow ((v_{i_S})^T g^T g a_{ij}^t, (v_{i_S})^T g^T g b_{ij}^t, (v_{i_S})^T g^T g c_{ij}^t) \\
&\quad = (v_{i_S} \cdot a_{ij}^t, v_{i_S} \cdot b_{ij}^t, v_{i_S} \cdot c_{ij}^t),
\end{aligned} \tag{B.15}
$$

where we used the fact that $g^T g = I$ due to the orthogonality of $g$ (with $I$ being the identity matrix). Therefore, the scalarization operation proposed in Equation B.14, and previously in Equation 3.3 in the main text (in an alternative form), yields SO(3)-invariant scalars, which is in line with the results of [29].

The output of Equation B.14, $q_{ij}$, is then aggregated in Equation 3.4 in the main text and concatenated in Equation 3.5 in the main text with the **GCP** module's remaining O(3)-invariant scalar features (i.e., $L_2$ vector norm features). Note that introducing SO(3)-invariant scalar information into the **GCP** module in this way breaks the 3D reflection symmetry that previous geometric graph convolution modules enforced [17], now giving rise within the **GCP** module to SO(3)-invariant and SO(3)-equivariant updates to scalar and vector-valued features, respectively. Therefore, scalar and vector-valued feature updates for nodes and edges within the **GCP** module are SO(3)-invariant and SO(3)-equivariant, respectively.

As in Appendix B.1.2, we now turn to discuss the operations within a single **GCPConv** layer, in particular proving that they maintain the respective SO(3) invariance and SO(3) equivariance for scalar and vector-valued features that the **GCP** module provides.

**SO(3)-equivariant GCPConv layer.** Via the corresponding proof in [17], by way of induction all such operations in Equations B.1-B.6 are respectively SE(3)-invariant and SO(3)-equivariant for features $m_{ij}^l = (m_{e_{ij}}^l, m_{\xi_{ij}}^l)$. Thereby, so are features $n_i^l = (h_i^l, \chi_i^l)$, given that the proof of equivariance for the equivariant **LayerNorm**

and **Dropout** operations employed within each **GCPConv** has previously been concretized by [17]. Equation B.8 concludes the operations of a single **GCPConv** layer by, as desired, updating the positions of each node $i$ in the 3D input graph. To do so, **GCPConv** residually updates current node positions $x_i^{l-1}$ using SO(3)-equivariant vector-valued features $\chi_{p_i}^l$. Therefore, **GCPConv** layers are SO(3)-invariant for scalar feature updates and SO(3)-equivariant for vector-valued node position and feature updates.

**SE(3)-equivariant GCPNet.** Lastly, as desired, Line 10 of Algorithm 1 in the main text adds $C(0)$ back to the predicted node positions $\mathbf{X}^l$ as provided by each **GCPConv** layer, ultimately imbuing position updates within $\mathbf{X}^l$ with SE(3) equivariance. Line 14 then concludes GCPNet by using the latest frame encodings $\boldsymbol{\mathcal{F}}_{ij}^t$ to perform, as desired, a final SO(3)-invariant and SO(3)-equivariant projection for scalar and vector-valued features, respectively. Therefore, as desired, GCPNets are SE(3)-invariant for scalar feature updates, SE(3)-equivariant for vector-valued node position and feature updates, and, as a consequence, satisfy the constraint proposed in Def. 1 of the main text.

□

### B.2.2 Proof of Proposition 2

*Proof.* The proof of SE(3) invariance for scalar node and edge features, $h_i$ and $e_{ij}$, follows as a corollary of Appendix B.2.1 (**SE(3)-equivariant GCPNet**). Therefore, GCPNets are SE(3)-invariant concerning their predicted scalar node and edge features and, as a consequence, are geometrically self-consistent according to the constraint in Def. 2 of the main text.

□

### B.2.3 Proof of Proposition 3

*Proof.* Suppose that GCPNet designates its local geometric representation for layer $t$ to be $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t)$, where $a_{ij}^t = \frac{x_i^t - x_j^t}{\|x_i^t - x_j^t\|}, b_{ij}^t = \frac{x_i^t \times x_j^t}{\|x_i^t \times x_j^t\|}$, and $c_{ij}^t = a_{ij}^t \times b_{ij}^t$, respectively. As in [77], this formulation of $\boldsymbol{\mathcal{F}}_{ij}^t$ is proven in Appendix B.2.1 (*SO(3)-equivariant frames*) to be an SO(3)-equivariant local orthonormal basis at the tangent space of $x_i^t$ and is thereby geometrically complete. Note this implies that GCPNet permits no loss of geometric information as discussed in Appendix A.5 of [77]. Therefore, GCPNets are geometry-complete and satisfy the constraint proposed in Def. 3 of the main text.

□

## B.3 IMPLEMENTATION DETAILS

**Featurization.**

Table B.1: Summary of GCPNet's node and edge features for 3D input graphs derived for the LBA and PSR tasks. Here, $N$ and $E$ denote the number of nodes and edges in $\mathcal{G}$, respectively.

| | Feature | Type | Shape |
|---|---|---|---|
| Node Features ($h$) | One-hot encoding of atom type | Categorical (Scalar) | $N \times 9$ |
| Node Features ($\chi$) | Directional encoding of orientation | Numeric (Vector) | $N \times 2$ |
| Edge Features ($e$) | Radial basis distance embedding | Numeric (Scalar) | $E \times 16$ |
| Edge Features ($\xi$) | Pairwise atom position displacement | Numeric (Vector) | $E \times 1$ |
| Total | Node features | | $N \times 11$ |
| | Edge features | | $E \times 17$ |

As shown in Table B.1, for the LBA and PSR tasks, in each 3D input graph, we include as a scalar node feature an atom's type using a 9-dimensional one-hot encoding vector for each atom. As vector-valued node features, we include *forward* and *reverse* unit vectors in the direction of $x_{i+1} - x_i$ and $x_{i-1} - x_i$, respectively (i.e., the node's 3D orientation). For the input 3D graphs' scalar edge features, we

Table B.2: Summary of GCPNet's node and edge features for 3D input graphs derived for the NMS task.

| | Feature | Type | Shape |
|---|---|---|---|
| Node Features ($h$) | Invariant velocity encoding | Numeric (Scalar) | $N \times 1$ |
| Node Features ($\chi$) | Velocity and orientation encoding | Numeric (Vector) | $N \times 3$ |
| Edge Features ($e$) | Edge and distance embedding | Numeric (Scalar) | $E \times 17$ |
| Edge Features ($\xi$) | Pairwise atom position displacement | Numeric (Vector) | $E \times 1$ |
| Total | Node features | | $N \times 4$ |
| | Edge features | | $E \times 18$ |

encode the distance $\|x_i - x_j\|_2$ using Gaussian radial basis functions, where we use 16 radial basis functions with centers evenly distributed between 0 and 20 units (e.g., Angstrom). For the graphs' vector-valued edge features, we encode the unit vector in the direction of $x_i - x_j$ (i.e., pairwise atom position displacements).

As illustrated in Table B.2, for the NMS task, in each 3D input graph, we include as a scalar node feature an invariant encoding of each node's velocity vector, namely $\sqrt{v_i^2}$. Each node's velocity and orientation are encoded as vector-valued node features. Scalar edge features are represented as Gaussian radial basis distance encodings as well as the product of the charges in each node pair (i.e., $c_i c_j$). Lastly, vector-valued edge features are represented as pairwise atom position displacements.

**Hardware used.** The Oak Ridge Leadership Facility (OLCF) at the Oak Ridge National Laboratory (ORNL) is an open science computing facility that supports HPC research. The OLCF houses the Summit compute cluster. Summit, launched in 2018, delivers 8 times the computational performance of Titan's 18,688 nodes, using only 4,608 nodes. Like Titan, Summit has a hybrid architecture, and each node contains multiple IBM POWER9 CPUs and NVIDIA Volta GPUs all connected with NVIDIA's high-speed NVLink. Each node has over half a terabyte of coherent memory (high bandwidth memory + DDR4) addressable by all CPUs and GPUs plus 800GB of non-volatile RAM that can be used as a burst buffer or as extended

memory. To provide a high rate of I/O throughput, the nodes are connected in a non-blocking fat-tree using a dual-rail Mellanox EDR InfiniBand interconnect. We used the Summit compute cluster to train all our models. For the LBA and NMS tasks, we used 16GB NVIDIA Tesla V100 GPUs for model training, whereas for the memory-intensive PSR and CPD tasks, we used 32GB V100 GPUs instead.

**Software used.** We used Python 3.8.12 [224], PyTorch 1.10.2 [225], PyTorch Lightning 1.7.7 [226], and PyTorch Geometric 2.1.0post0 [88] to run our deep learning experiments. For each model trained, PyTorch Lightning was used to facilitate model checkpointing, metrics reporting, and distributed data parallelism across 6 V100 GPUs. A more in-depth description of the software environment used to train and run inference with our models is available at `https://github.com/BioinfoMachineLearning/GCPNet`.

**Hyperparameters.** As shown in Tables B.3, B.4, B.5, and B.6, we use a learning rate of $10^{-4}$ with GCPNet for all tasks besides the RS task. The learning rate is kept constant throughout each model's training. For the NMS task, each model is trained for a minimum of 100 epochs and a maximum of 12,000 epochs. For all other tasks, each model is trained for a minimum of 100 epochs and a maximum of 1,000 epochs. For a given task, models with the best loss on the corresponding validation data split are then tested on the test split for the respective task. Note that, for the RS task, we do not perform any model hyperparameter tuning, following previous conventions from [83]. Test set run times are listed in Table B.7 for each task using the corresponding hyperparameter-tuned GCPNet model.

## B.4 REPRESENTATION LEARNING OF 3D BIOMOLECULES

### B.4.1 Comparison to existing protein representation learning methods

In Table B.8, we compare GCPNet to previous protein representation learning methods to highlight its distinguishing capabilities. In particular, GCPNet is the only

Table B.3: Hyperparameters used with all GCPNet models for the RS task.

| Hyperparameter | Selected Values |
|---|---|
| Number of GCPNet Layers | **8** |
| Number of **GCP** Message-Passing Layers | **8** |
| $\chi$ Hidden Dimensionality | **16** |
| Learning Rate | **0.0005** |
| Weight Decay Rate | **0** |
| **GCP** Dropout Rate | **0.1** |
| Dense Layer Dropout Rate | **0.1** |

Table B.4: Hyperparameter search space for all GCPNet models through which we searched to obtain strong performance on the LBA task's validation split. The final parameters for the standard GCPNet model for the LBA task are in **bold**.

| Hyperparameter | Search Space |
|---|---|
| Number of GCPNet Layers | 7, **8** |
| Number of **GCP** Message-Passing Layers | **8** |
| $\chi$ Hidden Dimensionality | **16**, 32 |
| Learning Rate | **0.0001**, 0.0003 |
| Weight Decay Rate | **0** |
| **GCP** Dropout Rate | **0.1**, 0.25 |
| Dense Layer Dropout Rate | **0.1**, 0.25 |

method that can produce arbitrary vector outputs while respecting SE(3) symmetries and, consequently, full sensitivity to molecular chirality. Furthermore, it can do so while representing 3D protein structures completely and self-consistently, thereby with no loss of force-related or geometric information.

### B.4.2 Future directions for representation learning of 3D biomolecules

Towards enhanced geometric representation learning of 3D biomolecules, we postulate that chirality sensitivity may be further strengthened for downstream tasks via a chirality-specific auxiliary loss function employed during model training. For example, drawing inspiration from AlphaFold 2 [15], such a loss function may periodically (e.g.,

Table B.5: Hyperparameter search space for all GCPNet models through which we searched to obtain strong performance on the PSR task's validation split. The final parameters for the standard GCPNet model for the PSR task are in **bold**.

| Hyperparameter | Search Space |
|---|---|
| Number of GCPNet Layers | **5** |
| Number of **GCP** Message-Passing Layers | **8** |
| $\chi$ Hidden Dimensionality | **16**, 32 |
| Learning Rate | **0.0001**, 0.0003 |
| Weight Decay Rate | **0**, 0.0001 |
| **GCP** Dropout Rate | **0.1**, 0.25 |
| Dense Layer Dropout Rate | **0.1**, 0.25 |

Table B.6: Hyperparameter search space for all GCPNet models through which we searched to obtain strong performance on the NMS task's validation split. The final parameters for the standard GCPNet model for the NMS task are in **bold**.

| Hyperparameter | Search Space |
|---|---|
| Number of GCPNet Layers | **4**, 7 |
| Number of **GCP** Message-Passing Layers | **8** |
| $\chi$ Hidden Dimensionality | **16** |
| Learning Rate | **0.0001**, 0.0003 |
| Weight Decay Rate | **0** |
| **GCP** Dropout Rate | 0.0, **0.1** |

50% of the time) penalize a method for producing scalar graph representations of mirrored biomolecular structures that are highly similar to one another in terms of cosine vector similarity, although doing so would require two forward passes of the model (e.g., 50% of the time) during training. Future work may investigate the utility of such auxiliary training objectives or efficient proxies of them.

To strengthen a method's awareness of global forces, we believe future research into optimal strategies for dynamically updating local coordinate frames during a method's forward pass may prove useful for geometric representation learning of atomic systems. For example, updating a method's local coordinate frames between individual network

Table B.7: Run times (in seconds) using GCPNet on the test dataset of each task.

| Task | Test Dataset Size | Run Time (s) |
|---|---|---|
| RS | 4480 | 472 |
| LBA | 490 | 47 |
| PSR | 16014 | 2231 |
| NMS - ES(5) | 2000 | 18 |
| NMS - ES(20) | 2000 | 12 |
| NMS - G+ES(20) | 2000 | 12 |
| NMS - L+ES(20) | 2000 | 12 |

Table B.8: Comparisons of existing protein geometric representation learning methods, adapted from [79]. Firstly, modeling protein graph nodes as atoms expands the range of molecular functions a method can directly represent (e.g., force field parameters) at the cost of increased computational complexity. Here $n$, $N$, and $k$ denote the number of amino acids, the number of atoms, and the average degree in a 3D protein graph, and $N >> n$. Our method is the only one that can learn and produce general vector outputs while maintaining SE(3) symmetries and thereby sensitivity to molecular chirality. Lastly, our method can do so while capturing 3D structures in a geometry-complete (i.e., local coordinates-wise) and self-consistent (i.e., scalar-wise SE(3)-invariant) manner.

| Method | Node Type | Complexity | Symmetry | Complete | Self-Consistent | Produces General Vectors | Chirality-Aware |
|---|---|---|---|---|---|---|---|
| GearNet [22] | Amino Acid | O(nk) | E(3) invariance | ✗ | ✗ | ✗ | ✗ |
| ProNet [79] | Amino Acid | O(nk) | SE(3) invariance | ✓ | ✓ | ✗ | ✓ |
| GVP-GNN etc. [17] | Amino Acid | O(nk) | E(3) equivariance | ✗ | ✗ | ✓ | ✗ |
| Vector-Gated GVP-GNN [76] | Atom | O(Nk) | E(3) equivariance | ✗ | ✗ | ✓ | ✗ |
| IEConv [229] | Atom | O(Nk) | E(3) invariance | ✗ | ✗ | ✗ | ✗ |
| ClofNet [77] | Atom | O(Nk) | SE(3) equivariance | ✓ | ✓ | ✗ | ✓ |
| Ours | Atom | O(Nk) | SE(3) equivariance | ✓ | ✓ | ✓ | ✓ |

layers that directly update node coordinates may yield fruitful results in this direction. Consequently, such techniques warrant further investigation in future work.

Lastly, in light of the promising results with GCPNet in Section 3.4 of the main text, future work on modeling could involve researching more computationally efficient variations of GCPNet that require fewer GCP message-passing layers within each GCP convolution layer or that embed geometric frames sparsely rather than in each GCP layer. For the LBA task in particular, incorporating known protein-ligand interaction information to predict binding affinity is a promising direction for future work on binding affinity prediction [80] and may lead to improved performance for

many of the methods listed in Table 2 of the main text. Another promising future direction to improve methods such as GCPNet is to improve the expressivity of such methods by learning higher-order equivariant tensors within one's message-passing procedure. Enhancing geometric expressiveness to thereby increase a method's effective run time efficiency would allow GCPNet to be used increasingly in new scientific and deep learning applications requiring high computational throughput (e.g., virtual screening of new drugs).

# Chapter C

# SUPPLEMENTARY MATERIALS FOR "GEOMETRY-COMPLETE DIFFUSION FOR 3D MOLECULE GENERATION AND OPTIMIZATION"

Adapted from Alex Morehead and Jianlin Cheng. "Geometry-complete diffusion for 3D molecule generation and optimization". *Communications Chemistry* 7.1 (2024): 150.

## C.1 SUPPLEMENTARY METHODS

### C.1.1 Expanded discussion of denoising

#### Geometry-complete denoising

In this section, we postulate that certain types of geometric neural networks serve as more effective 3D graph denoising functions for molecular DDPMs. We describe this notion as follows.

**Hypothesis C.1.1.** *(Geometry-Complete Denoising).*

*Geometric neural networks that achieve geometry-completeness are more robust in denoising 3D molecular network inputs compared to models that are not geometry-complete, in that geometry-complete methods unambiguously define direction-robust*

*local geometric reference frames.*

This hypothesis comes as an extension of the definition of geometry-completeness from [77] and [146]:

**Definition 4.** *(Geometric Completeness).*

*Given a pair of node positions* $(x_i^t, x_j^t)$ *in a 3D graph* $\mathcal{G}$*,*
*with vectors* $a_{ij}^t \in \mathbb{R}^{1\times 3}$*,* $b_{ij}^t \in \mathbb{R}^{1\times 3}$*, and* $c_{ij}^t \in \mathbb{R}^{1\times 3}$ *derived from* $(x_i^t, x_j^t)$*,*
*a local geometric representation* $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t) \in \mathbb{R}^{3\times 3}$ *is considered*
*geometrically complete if* $\boldsymbol{\mathcal{F}}_{ij}^t$ *is non-degenerate, hence forming*
*a local orthonormal basis located at the tangent space of* $x_i^t$*.*

An intuition for the implications of Hypothesis C.1.1 and Definition 4 on molecular diffusion models is that geometry-complete networks should be able to more effectively learn the gradients of data distributions [147] in which a global force field is present, as is typically the case with 3D molecules [77]. This is because, broadly speaking, geometry-complete methods encode local reference frames for each node (or edge) under which the directions of arbitrary global force vectors can be mapped. In addition to describing the theoretical benefits offered to geometry-complete denoising networks, we support this hypothesis through specific ablation studies in Sections 4.3.1 and 4.3.3 of the main text where we ablate the geometric frame encodings from GCDM and find that such frames are particularly useful in improving GCDM's ability to generate realistic 3D molecules.

**GCPNet++**

Inspired by its recent success in modeling 3D molecular structures with geometry-complete message-passing, we parametrize $p_{\mathbf{\Phi}}$ using an enhanced version of Geometry-Complete Perceptron Networks (GCPNets) that were originally introduced by [146]. To summarize, GCPNet is a geometry-complete graph neural network that is equivariant to SE(3) transformations of its graph inputs and maps nicely to the context of Hypothesis C.1.1.

In this setting, with ($h_i \in \mathbf{H}$, $\chi_i \in \boldsymbol{\chi}$, $e_{ij} \in \mathbf{E}, \xi_{ij} \in \boldsymbol{\xi}$), GCPNet++, our enhanced version of GCPNet, consists of a composition of Geometry-Complete Graph Convolution (**GCPConv**) layers $(h_i^l, \chi_i^l), x_i^l = \mathbf{GCPConv}[(h_i^{l-1}, \chi_i^{l-1}), (e_{ij}^{l-1}, \xi_{ij}^{l-1}), x_i^{l-1}, \mathcal{F}_{ij}]$ which are defined as:

$$n_i^l = \boldsymbol{\phi}^l(n_i^{l-1}, \mathcal{A}_{\forall j \in \mathcal{N}(i)} \mathbf{\Omega}_\omega^l(n_i^{l-1}, n_j^{l-1}, e_{ij}^{l-1}, \xi_{ij}^{l-1}, \mathcal{F}_{ij})), \tag{C.1}$$

where $n_i^l = (h_i^l, \chi_i^l)$; $\boldsymbol{\phi}^l$ is a trainable function; $l$ signifies the representation depth of the network; $\mathcal{A}$ is a permutation-invariant aggregation function; $\mathbf{\Omega}_\omega$ represents a message-passing function corresponding to the $\omega$-th **GCP** message-passing layer [146]; and node $i$'s geometry-complete local frames are $\mathcal{F}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t)$, with $a_{ij}^t = \frac{x_i^t - x_j^t}{\|x_i^t - x_j^t\|}, b_{ij}^t = \frac{x_i^t \times x_j^t}{\|x_i^t \times x_j^t\|}$, and $c_{ij}^t = a_{ij}^t \times b_{ij}^t$, respectively. Importantly, GCPNet++ restructures the network flow of **GCPConv** [146] for each iteration of node feature updates to simplify and enhance information flow, concretely from the form of

$$\hat{n}^l = n^{l-1} + f(\Omega_{\omega, v_i}^l | v_i \in \mathcal{V}) \tag{C.2}$$

to

$$\hat{n}^l = n^{l-1} \cup f((g_{e^\omega, v_i}^l, \Omega_{e^\omega, v_i}^l, \Omega_{\xi^\omega, v_i}^l) | v_i \in \mathcal{V}) \tag{C.3}$$

and from

$$n^l = \mathbf{ResGCP}^l_r(\tilde{n}^l_{r-1}) \tag{C.4}$$

to

$$n^l = \mathbf{GCP}^l_r(\tilde{n}^l_{r-1}). \tag{C.5}$$

Note that here $f$ represents a summation or a mean function that is invariant to node order permutations; $\cup$ denotes the concatenation operation; $g^l_{e^\omega,v_i}$ represents the binary-valued (i.e., [0, 1]) output of a scalar message attention (gating) function, expressed as

$$g^l_{e^\omega} = \sigma_{inf}(\phi^l_{inf}(\Omega^l_{e^\omega})) \tag{C.6}$$

with $\phi_{inf} : \mathbb{R}^e \to [0, 1]^1$ mapping from high-dimensional scalar edge feature space to a single dimension and $\sigma$ denoting a sigmoid activation function; $r$ is the node feature update module index; **ResGCP** is a version of the **GCP** module with added residual connections; and $\Omega^l_{\omega,v_i} = (\Omega^l_{e^\omega,v_i}, \Omega^l_{\xi^\omega,v_i})$ represents the scalar ($e$) and vector-valued ($\xi$) messages derived with respect to node $v_i$ using up to $\omega$ message-passing iterations within each GCPNet++ layer.

We found these adaptations to provide state-of-the-art molecule generation results compared to the original node feature updating scheme, which we found yielded sub-optimal results in the context of generative modeling. This highlights the importance of customizing representation learning algorithms for the generative modeling task at hand, since reasonable performance may not always be achievable with them without careful adaptations. It is worth noting that, since GCPNet++ performs message-passing directly on 3D vector features, GCDM is thereby the first diffusion generative model that is in principle capable of generating 3D molecules with specific vector-valued properties, thereby setting the stage for important future work.

### Properties of GCDM

If one desires to update the coordinate representations of each node in $\mathcal{G}$, as we do in the context of 3D molecule generation, the **GCPConv** module of GCPNet++ provides a simple, SE(3)-equivariant method to do so using a dedicated **GCP** module as follows:

$$(h_{p_i}^l, \chi_{p_i}^l) = \mathbf{GCP}_p^l(n_i^l, \mathcal{F}_{ij}) \tag{C.7}$$

$$x_i^l = x_i^{l-1} + \chi_{p_i}^l, \text{where } \chi_{p_i}^l \in \mathbb{R}^{1\times 3}, \tag{C.8}$$

where $\mathbf{GCP}_\cdot^l(\cdot, \mathcal{F}_{ij})$ is defined to provide chirality-aware rotation and translation-invariant updates to $h_i$ and rotation-equivariant updates to $\chi_i$ following centralization of the input point cloud's coordinates $\mathbf{X}$ [77]. The effect of using positional feature updates $\chi_{p_i}$ to update $x_i$ is, after decentralizing $\mathbf{X}$ following the final **GCPConv** layer, that updates to $x_i$ then become SE(3)-equivariant. As such, all transformations described above satisfy the required equivariance constraints. Therefore, in integrating GCPNet++ as its 3D graph denoiser, GCDM achieves SE(3) equivariance, geometry-completeness, and likelihood invariance altogether. Important to note is that GCDM subsequently performs message-passing with vector features to denoise its geometric inputs, whereas previous methods denoise their inputs solely using geometrically-insufficient scalar message-passing [107] as we demonstrate through our experiments in Section 4.3 of the main text.

### C.1.2 Expanded discussion of diffusion

#### Diffusion models

Key to understanding the contributions in this work are denoising diffusion probabilistic models (DDPMs). As alluded to previously, once trained, DDPMs can generate new data of arbitrary shapes, sizes, formats, and geometries by learning to reverse a noising process acting on each model input. More precisely, for a given data point $\mathbf{x}$, a diffusion process adds noise to $\mathbf{x}$ for time step $t = 0, 1, ..., T$ to yield $\mathbf{z}_t$, a noisy representation of the input $\mathbf{x}$ at time step $t$. Such a process is defined by a multivariate Gaussian distribution:

$$q(\mathbf{z}_t|x) = \mathcal{N}(\mathbf{z}_t|\alpha_t\mathbf{x}_t, \sigma_t^2\mathbf{I}), \tag{C.9}$$

where $\alpha_t \in \mathbb{R}^+$ regulates how much feature signal is retained and $\sigma_t^2$ modulates how much feature noise is added to input $\mathbf{x}$. Note that we typically model $\alpha$ as a function defined with smooth transitions from $\alpha_0 = 1$ to $\alpha_T = 0$, where a special case of such a noising process, the variance preserving process [230, 147], is defined by $\alpha_t = \sqrt{1 - \sigma_t^2}$. To simplify notation, in this work, we define the feature signal-to-noise ratio as $\text{SNR}(t) = \alpha_t^2/\sigma_t^2$. Also interesting to note is that this diffusion process is Markovian in nature, indicating that we have transition distributions as follows:

$$q(\mathbf{z}_t|\mathbf{z}_s) = \mathcal{N}(\mathbf{z}_t|\alpha_{t|s}\mathbf{z}_s, \sigma_{t|s}^2\mathbf{I}), \tag{C.10}$$

for all $t > s$ with $\alpha_{t|s} = \alpha_t/\alpha_s$ and $\sigma_{t|s}^2 = \sigma_t^2 - \alpha_{t|s}^2\sigma_s^2$. In total, then, we can write the noising process as:

$$q(\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_T|\mathbf{x}) = q(\mathbf{z}_0|x)\prod_{t=1}^{T} q(\mathbf{z}_t|\mathbf{z}_{t-1}). \tag{C.11}$$

If we then define $\boldsymbol{\mu}_{t\to s}(\mathbf{x}, \mathbf{z}_t)$ and $\sigma_{t\to s}$ as

$$\boldsymbol{\mu}_{t\to s}(\mathbf{x}, \mathbf{z}_t) = \frac{\alpha_{t|s}\sigma_s^2}{\sigma_t^2}\mathbf{z}_t + \frac{\alpha_s \sigma_{t|s}^2}{\sigma_t^2}\mathbf{x} \quad \text{and} \quad \sigma_{t\to s} = \frac{\sigma_{t|s}\sigma_s}{\sigma_t},$$

we have that the inverse of the noising process, the true denoising process, is given by the posterior of the transitions conditioned on $\mathbf{x}$, a process that is also Gaussian:

$$q(\mathbf{z}_s|\mathbf{x}, \mathbf{z}_t) = \mathcal{N}(\mathbf{z}_s|\boldsymbol{\mu}_{t\to s}(\mathbf{x}, \mathbf{z}_t), \sigma_{t\to s}\mathbf{I}). \tag{C.12}$$

**The Generative Denoising Process.** In diffusion models, we define the generative process according to the true denoising process. However, for such a denoising process, we do not know the value of $\mathbf{x}$ a priori, so we typically approximate it as $\hat{\mathbf{x}} = \phi(\mathbf{z}_t, t)$ using a neural network $\phi$. Doing so then lets us express the generative transition distribution $p(\mathbf{z}_s|\mathbf{z}_t)$ as $q(\mathbf{z}_s|\hat{\mathbf{x}}(\mathbf{z}_t, t), \mathbf{z}_t)$. As a practical alternative to Eq. C.12, we can represent this expression using the approximation for $\hat{\mathbf{x}}$:

$$p(\mathbf{z}_s|\mathbf{z}_t) = \mathcal{N}(\mathbf{z}_s|\boldsymbol{\mu}_{t\to s}(\hat{\mathbf{x}}, \mathbf{z}_t), \sigma_{t\to s}^2\mathbf{I}). \tag{C.13}$$

If we choose to define $s$ as $s = t - 1$, then we can derive the variational lower bound on the log-likelihood of $\mathbf{x}$ given the generative model as:

$$\log p(\mathbf{x}) \geq \mathcal{L}_0 + \mathcal{L}_{base} + \sum_{t=1}^{T} \mathcal{L}_t, \tag{C.14}$$

where we note that $\mathcal{L}_0 = \log p(\mathbf{x}|\mathbf{z}_0)$ models the likelihood of the data given its noisy representation $\mathbf{z}_0$, $\mathcal{L}_{base} = -\mathrm{KL}(q(\mathbf{z}_T|\mathbf{x})|p(\mathbf{z}_T))$ models the difference between a standard normal distribution and the final latent variable $q(\mathbf{z}_T|\mathbf{x})$, and

$$\mathcal{L}_t = -\mathrm{KL}(q(\mathbf{z}_s|\mathbf{x}, \mathbf{z}_t)|p(\mathbf{z}_s|\mathbf{z}_t)) \quad \text{for } t = 1, 2, ..., T.$$

Note that, in this formation of diffusion models, the neural network $\phi$ directly predicts $\hat{\mathbf{x}}$. However, [147] and others have found optimization of $\phi$ to be made much easier when instead predicting the Gaussian noise added to $\mathbf{x}$ to create $\hat{\mathbf{x}}$. An intuition for how this changes the neural network's learning dynamics is that, when predicting back the noise added to the model's input, the network is being trained to more directly differentiate which part of $\mathbf{z}_t$ corresponds to the input's feature signal (i.e., the underlying data point $\mathbf{x}$) and which part corresponds to added feature noise. In doing so, if we let $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$, the neural network can then predict $\hat{\boldsymbol{\epsilon}} = \phi(\mathbf{z}_t, t)$ such that:

$$\hat{\mathbf{x}} = (1/\alpha_t)\ \mathbf{z}_t - (\sigma_t/\alpha_t)\ \hat{\boldsymbol{\epsilon}}. \tag{C.15}$$

[215] and others have since shown that, when parametrizing the denoising neural network in this way, the loss term $\mathcal{L}_t$ reduces to:

$$\mathcal{L}_t = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\frac{1}{2}(1 - \mathrm{SNR}(t-1)/\mathrm{SNR}(t))\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}\|^2\right] \tag{C.16}$$

Note that, in practice, the loss term $\mathcal{L}_{base}$ should be close to zero when using a noising schedule defined such that $\alpha_T \approx 0$. Moreover, if and when $\alpha_0 \approx 1$ and $\mathbf{x}$ is a discrete value, we will find $\mathcal{L}_0$ to be close to zero as well.

### Zeroth likelihood terms for GCDM optimization objective

For the zeroth likelihood terms corresponding to each type of input feature, we directly adopt the respective terms previously derived by [114]. Doing so enables a negative log-likelihood calculation for GCDM's predictions. In particular, for integer node features, we adopt the zeroth likelihood term:

$$p(\mathbf{h}|\mathbf{z}_0^{(h)}) = \int_{\mathbf{h}-\frac{1}{2}}^{\mathbf{h}+\frac{1}{2}} \mathcal{N}(\mathbf{u}|\mathbf{z}_0^{(h)}, \sigma_0)\mathrm{d}\mathbf{u}, \tag{C.17}$$

where we use the CDF of a standard normal distribution, $\Phi$, to compute Eq. C.17 as $\Phi((\mathbf{h}+\frac{1}{2}-\mathbf{z}_0^{(h)})/\sigma_0)-\Phi((\mathbf{h}-\frac{1}{2}-\mathbf{z}_0^{(h)})/\sigma_0)\approx 1$ for reasonable noise parameters $\alpha_0$ and $\sigma_0$ [114]. For categorical node features, we instead use the zeroth likelihood term:

$$p(\mathbf{h}|\mathbf{z}_0^{(h)}) = C(\mathbf{h}|\mathbf{p}), \mathbf{p} \propto \int_{1-\frac{1}{2}}^{1+\frac{1}{2}} \mathcal{N}(\mathbf{u}|\mathbf{z}_0^{(h)}, \sigma_0)\mathrm{d}\mathbf{u}, \tag{C.18}$$

where we normalize $\mathbf{p}$ to sum to one and where $C$ is a categorical distribution [114]. Lastly, for continuous node positions, we adopt the zeroth likelihood term:

$$p(\mathbf{x}|\mathbf{z}_0^{(x)}) = \mathcal{N}\left(\mathbf{x}|\mathbf{z}_0^{(x)}/\alpha_0 - \sigma_0/\alpha_0\hat{\boldsymbol{\epsilon}}_0, \sigma_0^2/\alpha_0^2\mathbf{I}\right) \tag{C.19}$$

which gives rise to the log-likelihood component $\mathcal{L}_0^{(x)}$ as:

$$\mathcal{L}_0^{(x)} = \mathbb{E}_{\boldsymbol{\epsilon}^{(x)}\sim\mathcal{N}_x(\mathbf{0},\mathbf{I})}\left[\log Z^{-1} - \frac{1}{2}\|\boldsymbol{\epsilon}^x - \phi^{(x)}(\mathbf{z}_0, 0)\|^2\right], \tag{C.20}$$

where $d=3$ and the normalization constant $Z=(\sqrt{2\pi}\cdot\sigma_0/\alpha_0)^{(N-1)\cdot d}$ - in particular, its $(N-1)\cdot d$ term - arises from the zero center of gravity trick mentioned in Section 4.5.4 of the main text [114].

### Diffusion models and equivariant distributions

In the context of diffusion generative models of 3D data, one often desires for the marginal distribution $p(\mathbf{x})$ of their denoising neural network to be an invariant distribution. Towards this end, we observe that a conditional distribution $p(y|x)$ is equivariant to the action of 3D rotations by meeting the criterion:

$$p(y|x) = p(\mathbf{R}y|\mathbf{R}x) \quad \text{for all orthogonal } \mathbf{R}. \tag{C.21}$$

Moreover, a distribution is invariant to rotation transformations $\mathbf{R}$ when

$$p(y) = p(\mathbf{R}y) \quad \text{for all orthogonal } \mathbf{R}. \tag{C.22}$$

As [231] and [103] have collectively demonstrated, we know that if $p(\mathbf{z}_T)$ is invariant and the neural network we use to parametrize $p(\mathbf{z}_{t-1}|\mathbf{z}_t)$ is equivariant, we have, as desired, that the marginal distribution $p(\mathbf{x})$ of the denoising model is an invariant distribution.

### Training and sampling procedures for GCDM

**Equivariant Dynamics.** In this work, we use the previous definition of GCPNet++ in Section C.1.1 to learn an SE(3)-equivariant dynamics function $[\hat{\boldsymbol{\epsilon}}^{(x)}, \hat{\boldsymbol{\epsilon}}^{(h)}] = \phi(\mathbf{z}_t^{(x)}, \mathbf{z}_t^{(h)}, t)$ as:

$$\hat{\boldsymbol{\epsilon}}_t^{(x)}, \hat{\boldsymbol{\epsilon}}_t^{(h)} = \text{GCPNet++}(\mathbf{z}_t^{(x)}, [\mathbf{z}_t^{(h)}, \psi(\mathbf{z}_t^{(x)}), t/T]) - [\mathbf{z}_t^{(x)}, \mathbf{0}], \tag{C.23}$$

where we inform the denoising model of the current time step by concatenating $t/T$ as an additional node feature and where we subtract the coordinate representation outputs of GCPNet++ from its coordinate representation inputs after subtracting from the coordinate representation outputs their collective center of gravity. Lastly yet importantly, as a geometric GNN, GCPNet++ can embed geometric vector features in addition to scalar features. Subsequently, from the noisy coordinates representation $\mathbf{z}_t^{(x)}$ we derive noisy sequential (node) orientation unit vectors and pairwise (edge) displacement unit vectors $\psi(\mathbf{z}_t^{(x)})$, respectively, and embed these features using GCPNet++'s vector feature channels for nodes and edges accordingly. With the parametrization in Eq. 4.5 of the main text, GCDM subsequently achieves rotation equivariance on $\hat{\mathbf{x}}_i$, thereby achieving a 3D translation and rotation-invariant marginal distribution $p(\mathbf{x})$ as described in Appendix C.1.2.

**Scaling Node Features.** In line with [114], to improve the log-likelihood of the model's generated samples, we find it useful to train and perform sampling with GCDM using scaled node feature inputs as $[\mathbf{x}, \frac{1}{4}\mathbf{h}^{(categorical)}, \frac{1}{10}\mathbf{h}^{(integer)}]$.

**Deriving The Number of Atoms.** Finally, to determine the number of atoms with which GCDM will generate a 3D molecule, we first sample $N \sim p(N)$, where $p(N)$ denotes the categorical distribution of molecule sizes over GCDM's training dataset. Then, we conclude by sampling $\mathbf{x}, \mathbf{h} \sim p(\mathbf{x}, \mathbf{h}|N)$.

## C.2 SUPPLEMENTARY NOTES

### C.2.1 Broader impacts

In this chapter, we investigated the impact of geometric representation learning on generative models for 3D molecules. Such research can contribute to drug discovery efforts by accelerating the development of new medicinal or energy-related molecular compounds, and, as a consequence, can yield positive societal impacts [232]. Nonetheless, in line with [233], we authors would argue that it will be critical for institutions, governments, and nations to reach a consensus on the strict regulatory practices that should govern the use of such molecule design methodologies in settings in which it is reasonably likely such methodologies could be used for nefarious purposes by scientific "bad actors".

### C.2.2 Training details

**Scalar Message Attention.** In our implementation of scalar message attention (SMA) within GCDM, $\mathbf{m}_{ij} = e_{ij}\mathbf{m}_{ij}$, where $\mathbf{m}_{ij}$ represents the scalar messages learned by GCPNet++ during message-passing and $e_{ij}$ represents a 1 if an edge exists between nodes $i$ and $j$ (and a 0 otherwise) via $e_{ij} \approx \phi_{inf}(\mathbf{m}_{ij})$. Here, $\phi_{inf} : \mathbb{R}^e \to [0, 1]^1$ resembles a linear layer followed by a sigmoid function [18].

**GCDM hyperparameters.** All GCDM models train on QM9 for approximately 1,000 epochs using 9 **GCPConv** layers; SiLU activations [234]; 256 and 64 scalar node and edge hidden features, respectively; and 32 and 16 vector-valued node and edge features, respectively. All GCDM models are also trained using the AdamW optimizer [235] with a batch size of 64, a learning rate of $10^{-4}$, and a weight decay rate of $10^{-12}$.

**GCDM runtime.** With a maximum batch size of 64, this 9-layer model configuration allows us to train GCDM models for unconditional (conditional) tasks

on the QM9 dataset using approximately 10 (15) days of GPU training time with a single 24GB NVIDIA A10 GPU. For unconditional molecule generation on the much larger GEOM-Drugs dataset, a maximum batch size of 64 allows us to train 4-layer GCDM models using approximately 60 days of GPU training time with a single 48GB NVIDIA RTX A6000 GPU. As such, access to several GPUs with larger GPU memory limits (e.g., 80GBs) should allow one to concurrently train GCDM models in a fraction of the time via larger batch sizes or data-parallel training techniques [226].

### C.2.3 Compute requirements

Training GCDM models for tasks on the QM9 dataset by default requires a GPU with at least 24GB of GPU memory. Inference with such GCDM models for QM9 is much more flexible in terms of GPU memory requirements, as users can directly control how soon a molecule generation batch will complete according to the size of molecules being generated as well as one's selected batch size during sampling. Training GCDM models for unconditional molecule generation on the GEOM-Drugs dataset by default requires a GPU with at least 48GB of GPU memory. Similar to the GCDM models for QM9, inference with GEOM-Drugs models is flexible in terms of GPU memory requirements according to one's choice of sampling hyperparameters. Note that inference for both QM9 models and GEOM-Drugs models can likely be accelerated using techniques such as DDIM sampling [143]. However, we have not officially validated the quality of generated molecules using such sampling techniques, so we caution users to be aware of this potential risk of degrading molecule sample quality when using such sampling algorithms.

### C.2.4 Reproducibility

On GitHub, we thoroughly provide all source code, data, and instructions required to train new GCDM models or reproduce our results for each of the four protein-independent molecule generation tasks we study in this work. The source code, data, and instructions for our protein-conditional molecule generation experiments are also available on GitHub. Our source code uses PyTorch [236] and PyTorch Lightning [226] to facilitate model training; PyTorch Geometric [237] to support sparse tensor operations on geometric graphs; and Hydra [238] to enable reproducible hyperparameter and experiment management.

| Task<br>Units | $\alpha \downarrow$ / $MS \uparrow$<br>$Bohr^3$ / % | $\Delta\epsilon \downarrow$ / $MS \uparrow$<br>$meV$ / % | $\epsilon_{HOMO} \downarrow$ / $MS \uparrow$<br>$meV$ / % | $\epsilon_{LUMO} \downarrow$ / $MS \uparrow$<br>$meV$ / % | $\mu \downarrow$ / $MS \uparrow$<br>$D$ / % | $C_v \downarrow$ / $MS \uparrow$<br>$\frac{cal}{mol}K$ / % |
|---|---|---|---|---|---|---|
| Initial Samples (Moderately Stable) | $4.61 \pm 0.2$ / 61.7 | $1.26 \pm 0.1$ / 61.7 | $0.53 \pm 0.0$ / 61.7 | $1.25 \pm 0.0$ / 61.7 | $1.35 \pm 0.1$ / 61.7 | $2.93 \pm 0.1$ / 61.7 |
| EDM-Opt (100 steps on initial samples) | $4.45 \pm 0.6$ / $77.6 \pm 2.1$ | $0.98 \pm 0.1$ / $80.0 \pm 2.0$ | $0.45 \pm 0.0$ / $78.8 \pm 1.0$ | $0.91 \pm 0.0$ / $83.4 \pm 4.6$ | $6e^5 \pm 6e^5$ / $78.3 \pm 2.9$ | $2.72 \pm 2.6$ / $51.0 \pm 109.7$ |
| EDM-Opt (250 steps on initial samples) | $1e^2 \pm 5e^2$ / $80.1 \pm 2.1$ | $1e^3 \pm 6e^3$ / $83.7 \pm 3.8$ | $0.44 \pm 0.0$ / $82.5 \pm 1.3$ | $0.91 \pm 0.1$ / $\underline{84.7} \pm 1.6$ | $2e^5 \pm 8e^5$ / $\underline{81.0} \pm 5.8$ | $\underline{2.15} \pm 0.1$ / $\underline{78.5} \pm 3.4$ |
| GCDM-Opt (100 steps on initial samples) | $\underline{3.29} \pm 0.1$ / $\underline{86.2} \pm 1.3$ | $\underline{0.93} \pm 0.0$ / $\underline{89.0} \pm 1.9$ | $\mathbf{0.43} \pm 0.0$ / $\mathbf{91.6} \pm 3.5$ | $\underline{0.86} \pm 0.0$ / $\underline{87.0} \pm 1.7$ | $\underline{1.08} \pm 0.1$ / $\mathbf{89.9} \pm 4.2$ | $\mathbf{1.81} \pm 0.0$ / $\underline{87.6} \pm 1.1$ |
| GCDM-Opt (250 steps on initial samples) | $\mathbf{3.24} \pm 0.2$ / $\mathbf{86.6} \pm 1.9$ | $\mathbf{0.93} \pm 0.0$ / $\mathbf{89.7} \pm 2.2$ | $\underline{0.43} \pm 0.0$ / $\underline{90.7} \pm 0.0$ | $\mathbf{0.85} \pm 0.0$ / $\mathbf{88.6} \pm 3.8$ | $\mathbf{1.04} \pm 0.0$ / $\underline{89.5} \pm 2.6$ | $\underline{1.82} \pm 0.1$ / $\mathbf{87.6} \pm 2.3$ |

Table C.1: Comparison of GCDM with baseline methods for property-guided 3D molecule optimization. The results are reported in terms of molecular stability ($MS$) and the MAE for molecular property prediction by an ensemble of three EGNN classifiers $\phi_c$ (each trained on the same QM9 subset using a distinct random seed) yielding corresponding Student's t-distribution 95% confidence intervals, with results listed for EDM and GCDM-optimized samples as well as the molecule generation baseline ("Initial Samples"). Note that certain experiments with an EDM optimizer yielded unsuccessful property optimization, where we denote such results as outlier property MAE values greater than 50. The top-1 (best) results for this task are in **bold**, and the second-best results are underlined.

## C.3 SUPPLEMENTARY RESULTS

### C.3.1 Property-guided 3D molecule optimization - QM9

In Table C.1, for completeness, we list the numeric molecule optimization results comprising Figure 4.6 of the main text.

# Chapter D

# SUPPLEMENTARY MATERIALS FOR "GEOMETRIC FLOW MATCHING FOR GENERATIVE PROTEIN-LIGAND DOCKING AND AFFINITY PREDICTION"

Adapted from Alex Morehead and Jianlin Cheng. "FlowDock: Geometric Flow Matching for Generative Protein-Ligand Docking and Affinity Prediction". *Intelligent Systems for Molecular Biology & Bioinformatics* (ISMB 2025).

## D.1 GEOMETRIC FLOW MATCHING TRAINING AND INFERENCE

We characterize FlowDock's training and sampling procedures in Sections 5.3.5 (Training) and 5.3.6 (Sampling) of the main text, respectively. To further illustrate how training and inference with FlowDock work, in Algorithms 3 and 4 we provide the corresponding pseudocode. For more details, please see our accompanying source code at `https://github.com/BioinfoMachineLearning/FlowDock`.

**Algorithm 3** Training

**Require:** Training examples of binding site-aligned apo (holo) protein (ligand) structures, protein sequences, ligand SMILES strings, and binding affinities $\{(X^P_{a_i}, X^P_{h_i}, X^L_{h_i}, S_i, M_i, B_i)\}$

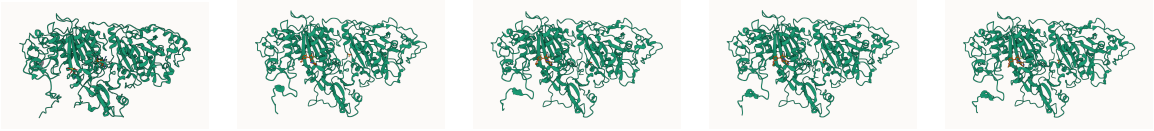
**for all** $(X^P_{a_i}, X^P_{h_i}, X^L_{h_i}, S_i, M_i, B_i)$ **do**
  Extract $x^P_1, x^L_1 \leftarrow \text{HeavyAtoms}(X^P_{h_i}, X^L_{h_i})$
  Sample $x^P_0 \leftarrow \text{ESMFold}(S_i) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma = 1e^{-4})$
  Sample $x^L_0 \leftarrow \text{HarmonicPrior}(M_{i_{frag}}), \quad \forall frag \in M_i$
  Sample $t \sim \mathcal{U}(0, 1)$
  Concatenate $x_0 = \text{Concat}(x^P_0, x^L_0)$
  Concatenate $x_1 = \text{Concat}(x^P_1, x^L_1)$
  Interpolate $x_t \leftarrow t \cdot x_1 + (1 - t) \cdot x_0$
  Predict $\hat{X}_{h_i} \leftarrow \text{NeuralPLexer}(S_i, M_i, x_t, t)$
  Predict $\hat{B}_i \leftarrow \text{ESDM}_{aff}(S_i, M_i, \text{StopGrad}(\hat{X}_{h_i}))$
  Optimize losses $\mathcal{L}_X := \lambda_X \cdot \text{FAPE}(X_{h_i}, \hat{X}_{h_i}) + \mathcal{L}_B := \lambda_B \cdot \text{MSE}(\hat{B}_i, B_i), \quad \lambda_X = 0.2, \quad \lambda_B = 0.1$

**Algorithm 4** Inference

**Require:** Protein sequences and ligand SMILES strings $(S, M)$

**Ensure:** Sampled top-5 heavy-atom structures $\hat{X}$ with confidence scores $\hat{C}$ and binding affinities $\hat{B}$

Sample $x^P_0 \leftarrow \text{ESMFold}(S) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma = 1e^{-4})$
Sample $x^L_0 \leftarrow \text{HarmonicPrior}(M_{frag}), \quad \forall frag \in M$
Concatenate $x_0 = \text{Concat}(x^P_0, x^L_0)$
**for** $n \leftarrow 0$ to $i$ **do**
  Let $t \leftarrow \frac{n}{i}$ and $s \leftarrow \frac{n+1}{i}$
  Predict $\hat{X} \leftarrow \text{NeuralPLexer}(S, M, x_n, t)$
  **if** $n = i - 1$ **then**
    Predict $\hat{C} \leftarrow \text{ESDM}_{conf}(S, M, \hat{X})$ # Pre-trained
    Predict $\hat{B} \leftarrow \text{ESDM}_{aff}(S, M, \hat{X})$
    Rank top-5 $\hat{X}$ and $\hat{B}$ using $\hat{C}$
    **return** $\hat{X}, \hat{C}, \hat{B}$
  Extract $\hat{x}_1 \leftarrow \text{HeavyAtoms}(\hat{X})$
  Align $x_n \leftarrow \text{RMSDAlign}(x_n, \hat{x}_1)$
  Interpolate $x_{n+1} = clamp(\frac{1-s}{1-t} \cdot \eta) \cdot x_n + clamp((1 - \frac{1-s}{1-t}) \cdot \eta) \cdot \hat{x}_1, \quad \eta = 1$



**FlowDock Init.** (t=1/40) | **FlowDock Interm.** (t=10/40) | **FlowDock Interm.** (t=20/40) | **FlowDock Interm.** (t=30/40) | **FlowDock Final** (t=40/40)

Figure D.1: Comparison of FlowDock's predicted structure states (w/o hydrogens) for CASP16 superligand pose pharma target L3008.

## D.2 STRUCTURE GENERATION EXAMPLE TRAJECTORY

To illustrate one of FlowDock's interpretable structure generation trajectories using conditional flow matching, in Figure D.1, we report FlowDock's predicted structural states for CASP16 superligand pose pharma target L3008, notably a *multi*-ligand pose target, in evenly spaced increments throughout FlowDock's generation trajectory. In short, we see that FlowDock enables multi-ligand protein complexes to be predicted through concise flow trajectories, yielding early protein and ligand conformational changes following the model's initial binding pocket prediction.

## D.3 CASP16 STRUCTURE PREDICTION RESULTS

In Figure D.2, we compare the protein-ligand structure prediction RMSDs of FlowDock and MULTICOM_ligand [239], a top-5 multi-model deep learning prediction method in the CASP16 ligand prediction category, for the 231 superligand pose pharma targets made available during the 16th Critical Assessment of Techniques for Structure Prediction (CASP16). As these results demonstrate, FlowDock, as a standalone deep learning method, achieves competitive structure predictions for many of the new CASP16 ligand targets. Similarly, Figure D.3 illustrates that FlowDock and MULTICOM_ligand are approximately tied in terms of their ability to structurally model CASP16's 56 *multi*-ligand protein complexes, highlighting the broad applicability of FlowDock's structure predictions in diverse drug discovery settings.

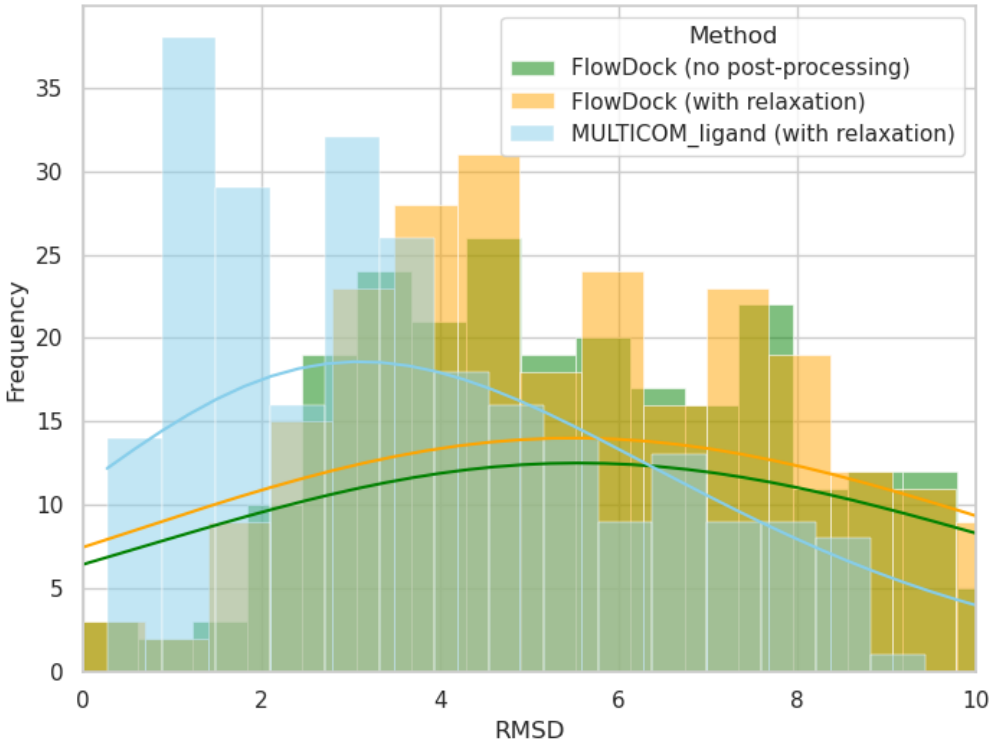



Figure D.2: Comparison of the protein-ligand structure prediction results of FlowDock and the deep learning ensembling method MULTICOM_ligand in terms of their binding pocket-aligned ligand RMSDs for the CASP16 superligand pose pharma targets (n=301).

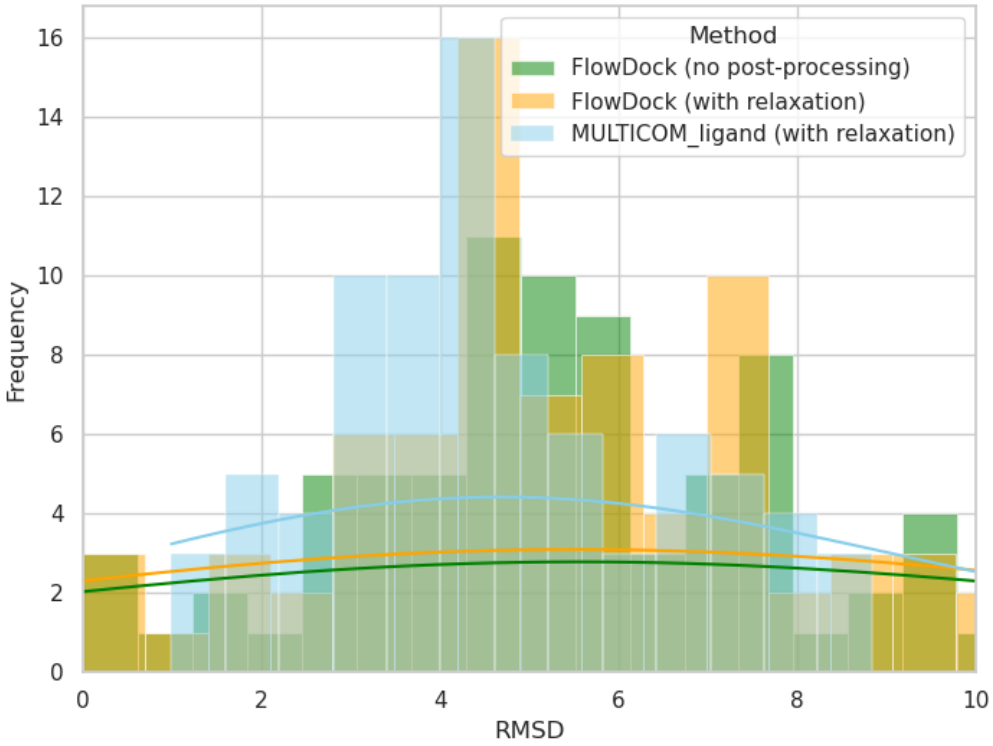



Figure D.3: Comparison of the protein-(multi-)ligand structure prediction results of FlowDock and the deep learning ensembling method MULTICOM_ligand in terms of their binding pocket-aligned ligand RMSDs for the CASP16 superligand pose pharma targets (n=126).

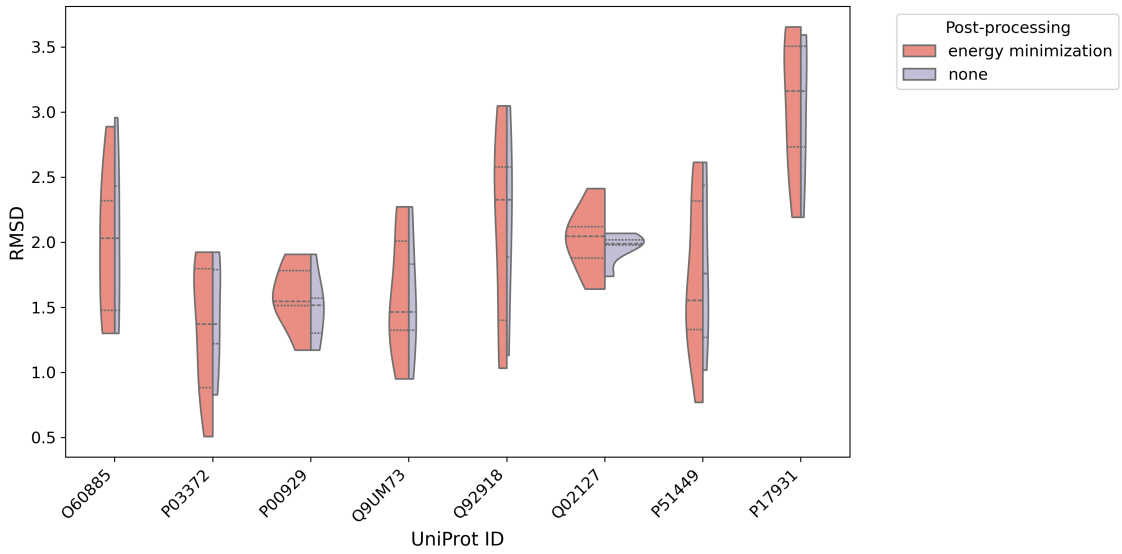



Figure D.4: Analysis of the protein-ligand structure prediction results of FLOWDOCK in terms of its binding pocket-aligned ligand RMSDs for the chemically dissimilar (multi-)ligand PoseBusters Benchmark targets (n=18).

## D.4 POSEBUSTERS BENCHMARK LIGAND DISSIMILARITY STRUCTURE PREDICTION RESULTS

To investigate FLOWDOCK's chemical generalization capabilities, in Figure D.4, we illustrate the structure prediction performance of FLOWDOCK for chemically dissimilar (Tanimoto similarity $< 0.6$) ligands associated with the same protein target in the PoseBusters Benchmark dataset. Figure D.4 shows that FLOWDOCK's average ligand RMSD of each of these (multi-)ligand protein targets is approximately 2 Å, with a standard deviation around 1 Å, highlighting that its predictions for chemically dissimilar intra-protein ligands are of high average accuracy and demonstrate generalizability with the consistency of FLOWDOCK's average inter-ligand RMSD differences.

# Chapter E

# SUPPLEMENTARY MATERIALS FOR "DEEP LEARNING FOR PROTEIN-LIGAND DOCKING: ARE WE THERE YET?"

Adapted from Alex Morehead, Nabin Giri, Jian Liu, Pawan Neupane, and Jianlin Cheng. "Deep Learning for Protein-Ligand Docking: Are We There Yet?". *AI for Science Workshop of the Forty-First International Conference on Machine Learning* (ICML 2024 AI4Science Spotlight).

## E.1 AVAILABILITY

The PoseBench codebase and tutorial notebooks are available under an MIT license at `https://github.com/BioinfoMachineLearning/PoseBench`. Preprocessed datasets and benchmark method predictions and results are available on Zenodo [240] under a CC-BY 4.0 license, of which the Astex Diverse and PoseBusters Benchmark datasets [112] and the DockGen-E dataset are associated with a CC-BY 4.0 license, and of which the CASP15 dataset [200], as a mixture of publicly and privately available resources, is partially licensed. In particular, 15 (4 single-ligand and 11 multi-ligand targets) of the 19 CASP15 protein-ligand interaction (PLI) complexes evaluated with PoseBench are publicly available, whereas the remaining 4 (2 single-ligand and 2 multi-ligand targets) are confidential and, for the purposes of future benchmarking and reproducibility, must be requested directly from the CASP organizers. Notably, the pre-holo-aligned protein structures predicted by AlphaFold

3 (AF3) for these four benchmark datasets (available on Zenodo [240]) must only be used in accordance with AF3's Terms of Use, whereas the pre-holo-aligned protein structures predicted by ESMFold for these four benchmark datasets (available on Zenodo [240]) are available under a permissive MIT license. Lastly, our use of the PoseBusters software suite for molecule validity checking is permitted under a BSD-3-Clause license.

### E.2 BROADER IMPACTS

Our benchmark unifies protein-ligand structure prediction datasets, methods, and tasks to enable enhanced insights into the real-world utility of such methods for accelerated drug discovery and energy research. We acknowledge the risk that, in the hands of "bad actors", such technologies may be used with harmful ends in mind. However, it is our hope that efforts in elucidating the performance of recent protein-ligand structure prediction methods in various macromolecular contexts will disproportionately influence the positive societal outcomes of such research such as improved medicines and subsequent clinical outcomes as opposed to possible negative consequences such as the development of new bioweapons.

### E.3 COMPUTE RESOURCES

To produce the results presented in this chapter, we ran a high performance computing sweep that concurrently utilized 12 80GB NVIDIA A100 GPUs for 14 days in total to run inference with each baseline method three times (where applicable), where each baseline deep learning (DL) method required approximately 24 hours of GPU compute to complete its inference runs (except for multiple sequence alignment (MSA)-dependent AF3 and RoseTTAFold-All-Atom (RFAA), which respectively took approximately 4 weeks and 2 weeks to finish their inference runs for each benchmark dataset). Notably, due to RFAA and AF3's significant storage requirements

Table E.1: The average runtime (in seconds) and peak memory usage (in GB) of each baseline method on a 25% subset of the Astex Diverse dataset (using an NVIDIA 80GB A100 GPU for benchmarking). The symbol - denotes a result that could not be estimated. Where applicable, an integer enclosed in parentheses indicates the number of samples drawn from a particular baseline method.

| Method | Runtime (s) | CPU Memory Usage (GB) | GPU Memory Usage (GB) |
|---|---|---|---|
| P2Rank-Vina (40) | 1,283.70 | 9.62 | 0.00 |
| DiffDock-L (5) | 88.33 | 8.99 | 70.42 |
| DynamicBind (5) | 146.99 | 5.26 | 18.91 |
| NeuralPLexer (5) | 29.10 | 11.19 | 31.00 |
| RoseTTAFold-All-Atom (1) | 3,443.63 | 55.75 | 72.79 |
| Chai-1 (5) | 114.86 | 58.49 | 56.21 |
| AF3 (5) | 3,049.41 | - | - |

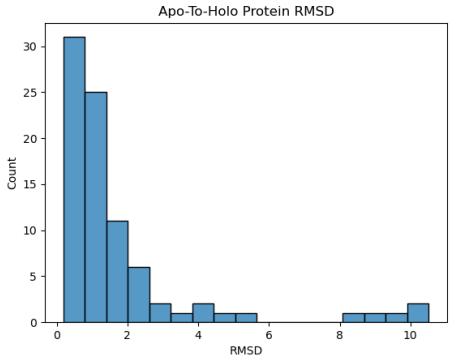
for running inference with their MSA databases, we utilized approximately 6 TB of solid-state storage space in total to benchmark all baseline methods. Lastly, in terms of CPU requirements, our experiments utilized approximately 64 concurrent CPU threads for AutoDock Vina inference (as an upper bound) and 60 GB of CPU RAM. Note that an additional 4-5 weeks of compute were spent performing initial (non-sweep) versions of each experiment during PoseBench's initial phase of development.

As a more formal investigation of the computational resources required to run each baseline method in this work, in Table E.1 we list the average runtime (in seconds) and peak CPU (GPU) memory usage (in GB) consumed by each method when running them on a 25% subset of the Astex Diverse dataset. We find that NeuralPLexer provides the lowest computational runtime and DynamicBind the lowest peak CPU and GPU memory requirements during benchmarking.

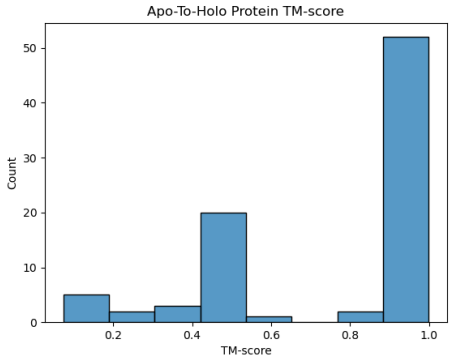
### E.4 DOCUMENTATION FOR DATASETS

Below, we provide detailed documentation for each dataset included in our benchmark, summarized in Table 1 of the main text. Each dataset is freely available for

download from the benchmark's accompanying Zenodo data record [240] under a CC-BY 4.0 license. In lieu of being able to create associated metadata for each of our macromolecular datasets using an ML-focused library such as Croissant [241] (due to file type compatibility issues), instead, we report structured metadata for our preprocessed datasets using Zenodo's web user interface [240]. Note that, for all datasets, we authors bear all responsibility in case of any violation of rights regarding the usage of such datasets.

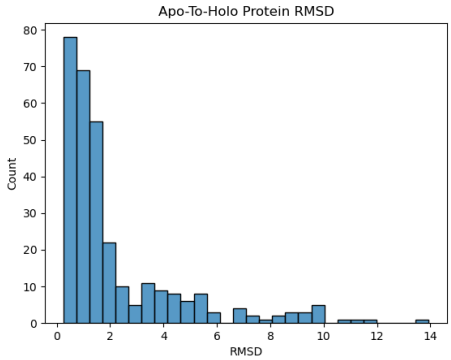



(a) RMSD of AF3's predictions.

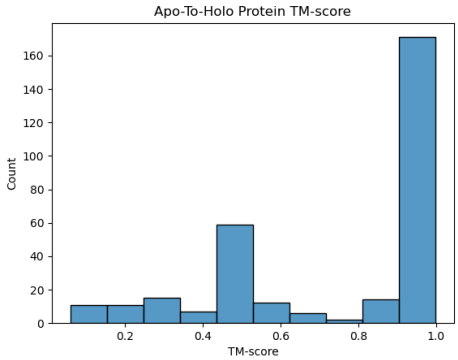



(b) TM-score of AF3's predictions.

Figure E.1: Accuracy of AF3's predicted protein structures for the Astex Diverse dataset.

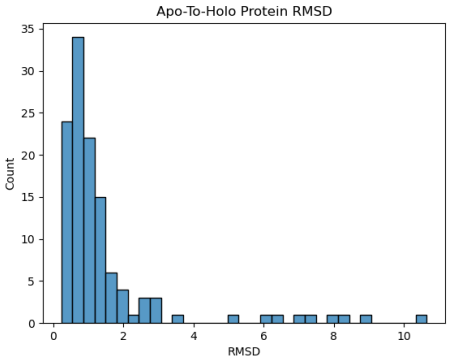



(a) RMSD of AF3's predictions.

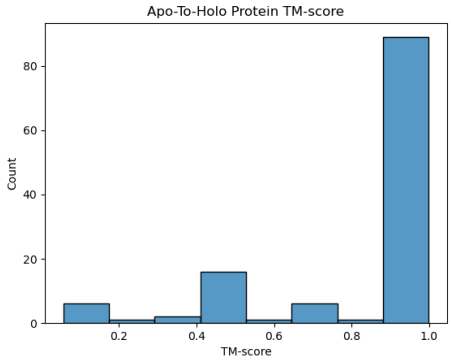



(b) TM-score of AF3's predictions.

Figure E.2: Accuracy of AF3's predicted protein structures for the PoseBusters Benchmark dataset.

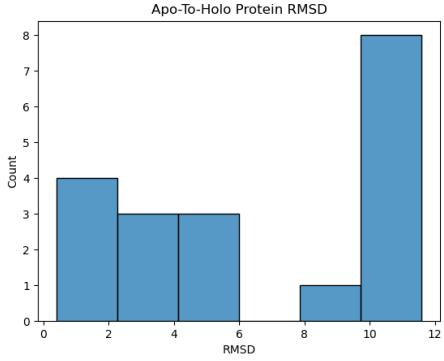



(a) RMSD of AF3's predictions.

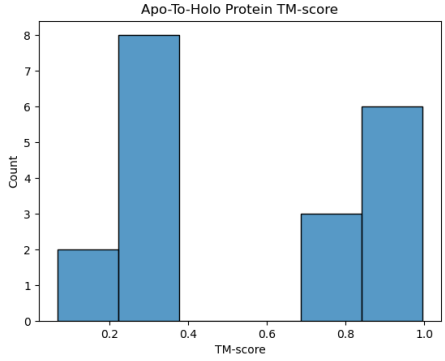



(b) TM-score of AF3's predictions.

Figure E.3: Accuracy of AF3's predicted protein structures for the DockGen dataset.




(a) RMSD of AF3's predictions.




(b) TM-score of AF3's predictions.

Figure E.4: Accuracy of AF3's predicted protein structures for the CASP15 dataset.

### E.4.1 Astex Diverse Set - Primary Ligand Docking (Difficulty: *Easy*)

A common drug discovery task is to screen several novel drug-like molecules against a target protein in rapid succession. The Astex Diverse dataset was originally developed with this application in mind, as it features many therapeutically relevant 3D molecules for computational modeling.

- **Motivation** Several downstream drug discovery efforts rely on having access to high-quality molecular data for docking.
- **Collection** For this dataset, which was originally compiled by [177], we adopt the version further prepared by [112].
- **Composition** The dataset consists of 85 primary ligand protein complexes deposited in the PDB up to 2007. As such, this dataset can be considered an easy benchmarking dataset since many of its complexes may be found in DL methods' PDB-based training datasets. For each of these complexes, we obtained high-accuracy predicted protein structures using AF3. The accuracy of the AF3-predicted structures is measured in terms of their RMSD and TM-score [171] compared to the corresponding crystal protein structures and is visualized in Figure E.1. Notably, after alignment with the crystalized (holo) PLI binding pocket residues, 63.53% (54.12% with ESMFold) of the predicted structures have a global RMSD below 4 Å and TM-score above 0.7, indicating that most of the dataset's proteins have a reasonably accurate predicted structure.
- **Hosting** Our preprocessed version of the dataset (`https://doi.org/10.5281/zenodo.14629652`) can be downloaded from the benchmark's Zenodo data record at `https://zenodo.org/records/14629652/files/astex_diverse_set.tar.gz`.

- **Licensing** We have released our preprocessed version of the dataset under a CC-BY 4.0 license. The original PoseBusters Benchmark dataset is available under a CC-BY 4.0 license on Zenodo [242]. The pre-holo-aligned protein structures predicted by AF3 for this dataset (available on Zenodo [240]) must only be used in accordance with AF3's Terms of Use.

- **Maintenance** We will announce any errata discovered in or changes made to the dataset using the benchmark's GitHub repository at `https://github.com/BioinfoMachineLearning/PoseBench`.

- **Uses** This dataset of predicted (apo) and crystal (holo) protein PDB and crystal (holo) ligand SDF files can be used for primary ligand docking or protein-ligand structure prediction.

- **Metrics** Ligand Centroid RMSD $\leq 1$ Å, Ligand Pose RMSD $\leq 2$ Å, PoseBusters-Valid (PB-Valid), and PLIF-WM.

### E.4.2 PoseBusters Benchmark Set - Primary Ligand Docking (Difficulty: *Intermediate*)

Like the Astex Diverse dataset, the PoseBusters Benchmark dataset was originally developed for docking individual ligands to target proteins. However, this dataset features a larger and more challenging collection of PLI complexes for computational modeling.

- **Motivation** Data sources of challenging primary ligand protein complexes for molecular docking are critical for the development of future docking methods.
- **Collection** For this dataset, we adopt the version introduced by [112].
- **Composition** The dataset consists of 308 primary ligand protein complexes deposited in the PDB in 2019 and after. As such, this dataset poses a moderate challenge for DL methods, since several of such methods were trained on data deposited before this cutoff date (notably except for Chai-1 and AF3 which used training cutoff dates of January 12, 2021 and September 30, 2021, respectively). For each of the dataset's complexes, we obtained high-accuracy predicted protein structures using AF3. The accuracy of the AF3-predicted structures is measured in terms of their RMSD and TM-score compared to the corresponding crystal protein structures and is visualized in Figure E.2. Notably, after alignment with the crystalized (holo) PLI binding pocket residues, 59.09% (53.25% with ESMFold) of the predicted structures have a global RMSD below 4 Å and TM-score above 0.7, indicating that most of the dataset's proteins have a reasonably accurate predicted structure.
- **Hosting** Our preprocessed version of the dataset (`https://doi.org/10.5281/zenodo.14629652`) can be downloaded from the benchmark's Zenodo data record at `https://zenodo.org/records/14629652/files/posebusters_benchmark_`

`set.tar.gz`.

- **Licensing** We have released our preprocessed version of the dataset under a CC-BY 4.0 license. The original dataset is available under a CC-BY 4.0 license on Zenodo [242]. The pre-holo-aligned protein structures predicted by AF3 for this dataset (available on Zenodo [240]) must only be used in accordance with AF3's Terms of Use.

- **Maintenance** We will announce any errata discovered in or changes made to the dataset using the benchmark's GitHub repository at `https://github.com/BioinfoMachineLearning/PoseBench`.

- **Uses** This dataset of predicted (apo) and crystal (holo) protein PDB and crystal (holo) ligand SDF files can be used for primary ligand docking or protein-ligand structure prediction.

- **Metrics** Ligand Centroid RMSD $\leq 1$ Å, Ligand Pose RMSD $\leq 2$ Å, PoseBusters-Valid (PB-Valid), and PLIF-WM.

### E.4.3 DockGen-E Set - Primary Ligand Docking (Difficulty: *Challenging*)

The DockGen dataset was originally designed for binding individual ligands to target proteins within functionally novel PLI binding pockets [181], filtering out any protein chains not associated with a novel pocket, which can remove important biomolecular context for DL methods to make their predictions. In this work, we introduced DockGen-E, an enhanced version of DockGen that has each method predict the full biologically relevant assembly of each novel pocket to expand their structural prediction contexts (n.b., which is specifically important to achieve best performance with DL co-folding methods such as AF3). As such, this new dataset is useful for evaluating how well each baseline method can predict complexes containing functionally distinct binding pockets compared to those on which the method may have *primarily* been trained.

- **Motivation** Data sources of PLI complexes representing novel primary ligand binding pockets are critical for the development of generalizable docking methods.

- **Collection** To curate this dataset, we collected the original dataset's protein and ligand binding pocket annotations for each complex introduced by [181]. Subsequently, we retrieved the corresponding first biological assembly listed in the PDB to obtain each PDB entry's biologically relevant complex, filtering out complexes for which the first assembly could not be mapped to its original protein and ligand binding pocket annotation. This procedure left 122 biologically relevant assemblies remaining for benchmarking. Important to note is that these original DockGen complexes were deposited in the PDB from 2019 onward, making this benchmarking dataset partially overlap with the training datasets of multiple DL co-folding baseline methods such as NeuralPLexer, AF3, and

Chai-1. Nonetheless, our benchmarking results in the main text demonstrate that baseline DL methods are challenged to find the correct (novel) binding pocket conformations represented by this dataset, suggesting that all baseline DL models have yet to learn truly comprehensive representations of protein-ligand binding.

- **Composition** The dataset consists of 122 primary ligand protein complexes, for each of which we obtained high-accuracy predicted protein structures using AF3. The accuracy of the AF3-predicted structures is measured in terms of their RMSD and TM-score compared to the corresponding crystal protein structures and is visualized in Figure E.3. Notably, after alignment with the crystalized (holo) PLI binding pocket residues, 74.59% (57.38% with ESMFold) of the predicted structures have a global RMSD below 4 Å and TM-score above 0.7, indicating that most of the dataset's proteins have a reasonably accurate predicted structure.

- **Hosting** Our preprocessed version of the dataset (`https://doi.org/10.5281/zenodo.14629652`) can be downloaded from the benchmark's Zenodo data record at `https://zenodo.org/records/14629652/files/dockgen_set.tar.gz`.

- **Licensing** We have released our preprocessed version of the DockGen-E dataset under a CC-BY 4.0 license. The original DockGen dataset is available under an MIT license on Zenodo [181], and the DockGen-E dataset along with its pre-holo-aligned protein structures predicted by AF3 is also available on Zenodo [240]. Notably, these AF3-predicted protein structures must only be used in accordance with AF3's Terms of Use.

- **Maintenance** We will announce any errata discovered in or changes made to the dataset using the benchmark's GitHub repository at `https://github.com/BioinfoMachineLearning/PoseBench`.

- **Uses** This dataset of predicted (apo) and crystal (holo) protein PDB and crystal (holo) ligand PDB files can be used for primary ligand docking or protein-ligand structure prediction.
- **Metrics** Ligand Centroid RMSD $\leq 1$ Å, Ligand Pose RMSD $\leq 2$ Å, PoseBusters-Valid (PB-Valid), and PLIF-WM.

### E.4.4 CASP15 Set - Multi-Ligand Docking

**(Difficulty: *Challenging*)**

As the most distinct of our benchmark's four evaluation datasets, the CASP15 PLI dataset was created to represent the new protein-ligand modeling category in the 15th Critical Assessment of Techniques for Structure Prediction (CASP) competition. Whereas datasets such as PoseBusters Benchmark and Astex Diverse feature solely primary ligand protein complexes, the CASP15 dataset provides the research community with a variety of challenging organic (e.g., drug molecules) and inorganic (e..g., ion) cofactors for *multi*-ligand biomolecular modeling and scoring.

- **Motivation** Multi-ligand evaluation datasets for molecular docking provide the rare opportunity to assess how well baseline methods can model intricate PLIs while avoiding troublesome inter-ligand steric clashes. Additionally, accurate modeling of multi-ligand complexes in future work may lead to improved algorithms for computational enzyme design and regulation [167].

- **Collection** For this dataset, we manually collect each publicly and privately available CASP15 protein-bound ligand complex structure compatible with protein-ligand (e.g., non-nucleic acid) benchmarking.

- **Composition** The dataset consists of 102 (86) fragment ligands contained within 19 (15) separate (publicly available) protein complexes, of which 6 (2) and 13 (2) of these complexes are single and multi-ligand complexes, respectively. Importantly, each of such complexes (if publicly available) was released in the PDB after 2022, making this benchmarking dataset strictly non-overlapping with the training datasets of all baseline methods. The accuracy of the dataset's AF3-predicted structures is measured in terms of their RMSD and TM-score compared to the corresponding crystal protein structures and is visualized in Figure E.4. Notably, after alignment with the crystalized (holo) PLI binding

pocket residues, 36.84% and 20.00% (26.32% and 13.33% with ESMFold) of the total and publicly available predicted structures, respectively, have a global RMSD below 4 Å and TM-score above 0.7, indicating that a portion of the dataset's proteins has a reasonably accurate predicted structure. Given the much larger structural assemblies of this dataset's protein complexes compared to those of the other benchmark datasets, we believe the accuracy of these predictions may be improved with advancements in machine learning modeling of biomolecular assemblies.

- **Hosting** Our preprocessed version of (the publicly available version of) this dataset (`https://doi.org/10.5281/zenodo.14629652`) can be downloaded from the benchmark's Zenodo data record at `https://zenodo.org/records/14629652/files/casp15_set.tar.gz`.

- **Licensing** We have released our preprocessed version of the (public) dataset under a CC-BY 4.0 license. The original (public) dataset is free for download via the RCSB PDB [173], whereas the dataset's remaining (private) complexes must be manually requested from the CASP organizers. The pre-holo-aligned protein structures predicted by AF3 for this dataset (available on Zenodo [240]) must only be used in accordance with AF3's Terms of Use.

- **Maintenance** We will announce any errata discovered in or changes made to the dataset using the benchmark's GitHub repository at `https://github.com/BioinfoMachineLearning/PoseBench`.

- **Uses** This dataset of predicted (apo) and crystal (holo) protein PDB and crystal (holo) ligand PDB files can be used for multi-ligand docking or protein-ligand structure prediction.

- **Metrics** (Fragment) Ligand Pose RMSD $\leq$ 2 Å, (Complex) PoseBusters-Valid

(PB-Valid), and (Complex) PLIF-WM.

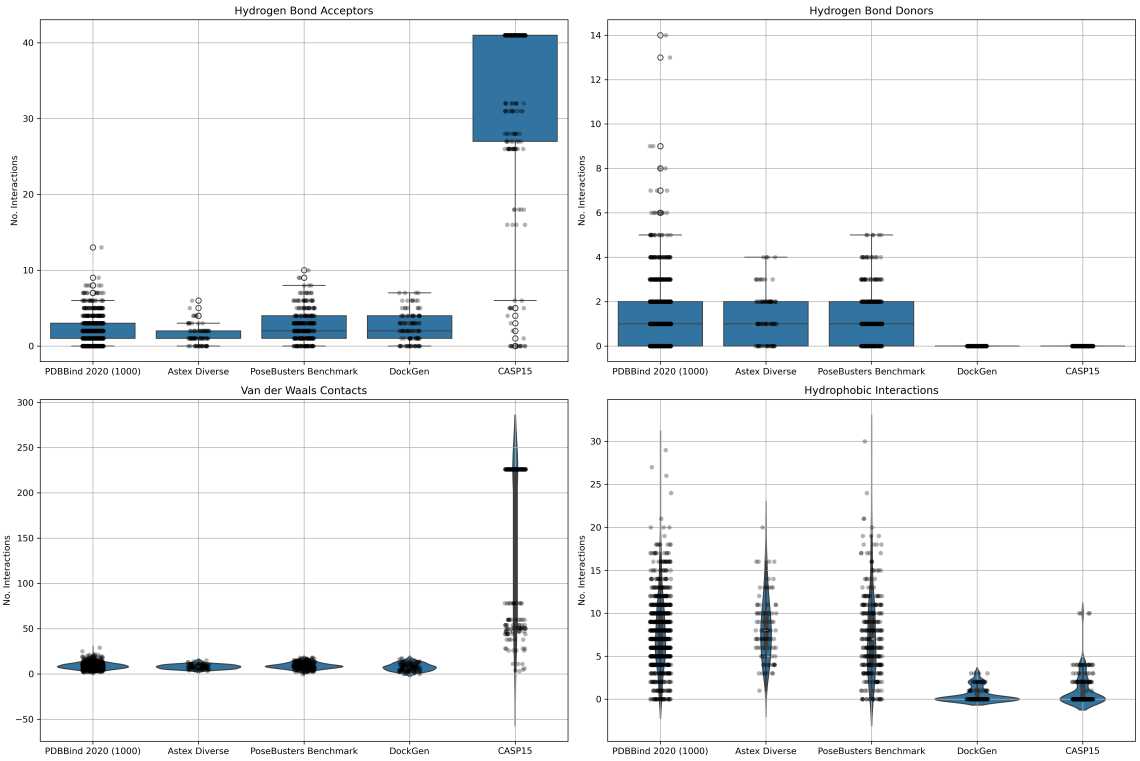



Figure E.5: Comparative analysis of evaluation dataset protein-ligand interactions.

## E.5 ANALYSIS OF PROTEIN-LIGAND INTERACTIONS

### E.5.1 Dataset protein-ligand interaction distributions

Inspired by a similar analysis presented in the PoseCheck benchmark [145], in this section, we study the frequency of different types of protein-ligand (pocket-level) interactions such as van der Waals contacts and hydrophobic interactions occurring natively within (n.b., a size-1000 random subset of) the commonly-used PDBBind 2020 docking training dataset (i.e., PDBBind 2020 (1000)) as well as the Astex Diverse, PoseBusters Benchmark, DockGen, and CASP15 benchmark datasets, respectively. In particular, these measures allow us to better understand the diversity of interactions each baseline method within the PoseBench benchmark is tasked to model, within the context of each evaluation dataset. Furthermore, these measures directly indicate which benchmark datasets are most *dissimilar* from commonly used training

data for baseline methods. Figure E.5 displays the results of this analysis.

Overall, we find that the PDBBind 2020, Astex Diverse, and PoseBusters Benchmark datasets contain similar types and frequencies of interactions, with the PoseBusters Benchmark dataset containing slightly more hydrogen bond acceptors (∼3 vs 1) and fewer van der Waals contacts (∼5 vs 8) on average compared to the PDBBind 2020 dataset. However, we observe a more notable difference in interaction types and frequencies between the DockGen and CASP15 datasets and the three other datasets. Specifically, we find these two benchmark datasets contain a notably different quantity of hydrogen bond acceptors and donors (n.b., ∼40 for CASP15), van der Waals contacts (∼200 for CASP15), and hydrophobic interactions (∼2 for DockGen) on average. These dataset-level interaction disparities may partially explain the baseline-challenging DockGen benchmarking results reported in Section 2 of the main text.

Also particularly interesting to note is the CASP15 dataset's bimodal distribution of van der Waals contacts, suggesting that the dataset contains two primary classes of interacting ligands giving rise to van der Waals interactions. One possible explanation for this phenomenon is that the CASP15 prediction targets, in contrast to the PDBBind, Astex Diverse, PoseBusters Benchmark, and DockGen targets, consist of a variety of both organic (e.g., drug-like molecules) and inorganic (e.g., metal) cofactors.

### E.5.2 Baseline method protein-ligand interaction distributions

Intrigued by the dataset interaction patterns presented in Figure E.5, here we further investigate the predicted PLIs produced by each baseline method for each evaluation dataset to study which DL methods can most faithfully reproduce the native distribution of PLIs within each dataset. Our results in Figures E.6, E.7, E.8, and E.9 suggest that AF3 demonstrates the best overall ability to recapitulate the crystalized

PLIs observed within these datasets, in line with the PLIF-WM benchmarking results presented in Section 2 of the main text. Nonetheless, its predicted interaction distributions, in particular for the DockGen and CASP15 datasets, have much room for improvement, especially for more structured interactions such as hydrogen bonds.

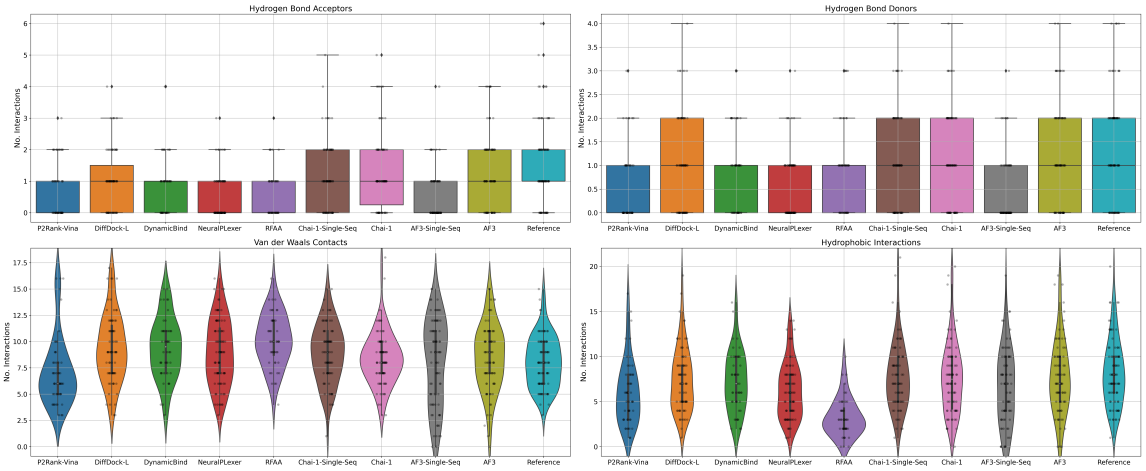



Figure E.6: Comparative analysis of Astex Diverse protein-ligand interactions.

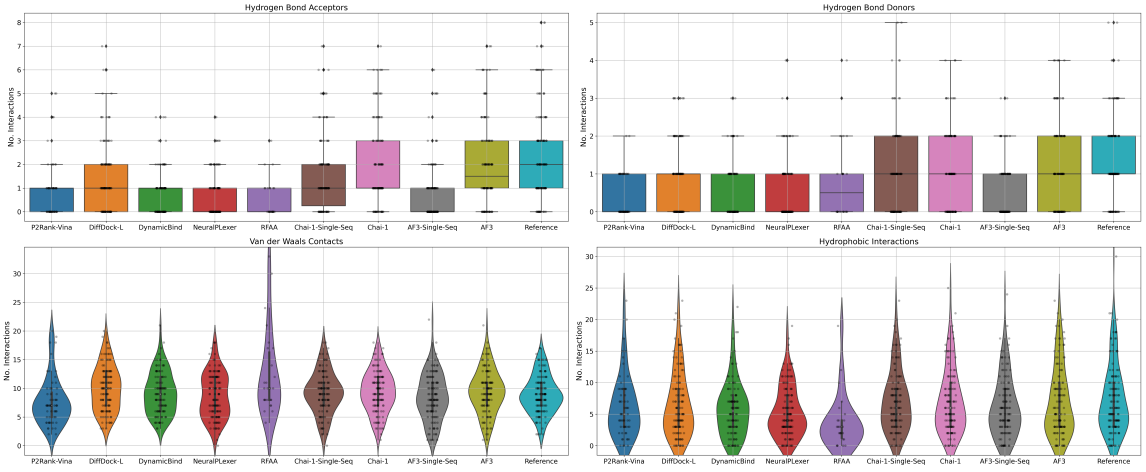



Figure E.7: Comparative analysis of PoseBusters Benchmark protein-ligand interactions.

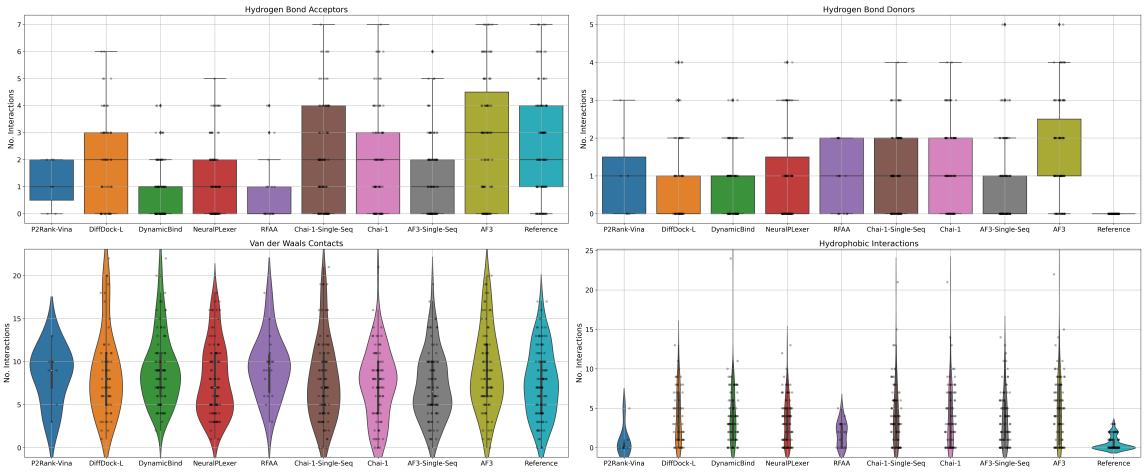



Figure E.8: Comparative analysis of DockGen protein-ligand interactions.

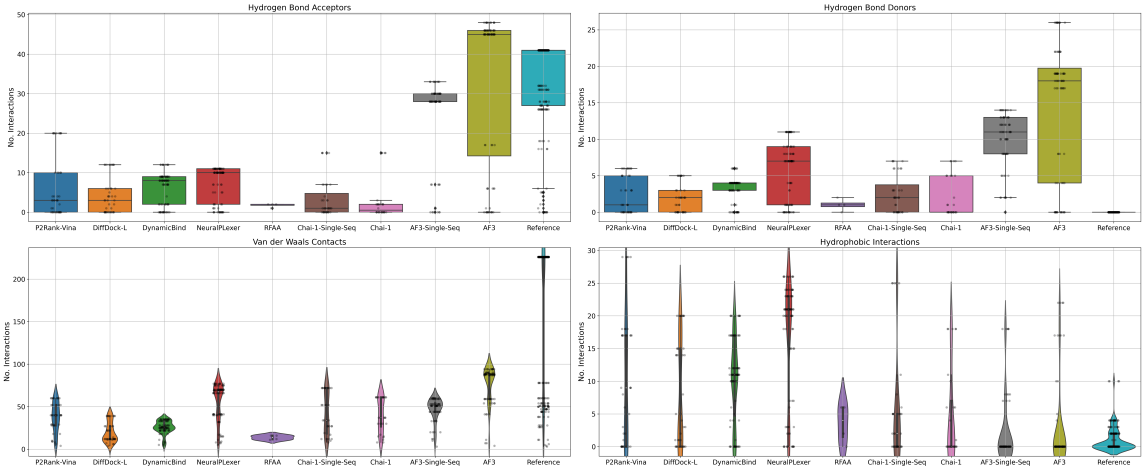



Figure E.9: Comparative analysis of CASP15 protein-ligand interactions.

## E.6 ADDITIONAL METHOD DESCRIPTIONS

To better contextualize the benchmark's results comparing DL methods to conventional docking algorithms, in this section, we provide further details regarding how each baseline method in the benchmark leverages different sources of biomolecular information to predict PLIs for a given protein target.

### E.6.1 Input and output formats

1. Formats for conventional methods are as follows:

    a) Molecular docking (protein-fixed) software tools such as **AutoDock Vina**, which require specification of protein binding sites, are provided with not only a predicted protein structure from AF3 but also the centroid coordinates of each predicted PLI binding site residue as estimated by the well-known P2Rank binding site prediction algorithm [211]. Such binding site residues are classified using a 10 Å protein-ligand heavy atom interaction threshold and a 25 Å inter-ligand heavy atom interaction threshold to define a "group" of ligands belonging to the same binding site and therefore residing in the same 25 $Å^3$-sized binding site input voxel for AutoDock Vina. For interested readers, for all four benchmark datasets, we also provide the benchmarking code necessary to run AutoDock Vina using any other baseline method's predicted binding site residues (e.g., those of DiffDock-L) according to the same binding site classification scheme described above.

2. Formats for DL docking methods are as follows:

    a) **DiffDock-L** is provided with a protein structure predicted by AF3 and (fragment) ligand SMILES strings. The model is then tasked with pro-

ducing (multiple rank-ordered) ligand conformations (for each fragment) for the given protein structure (which remains fixed during docking). Note that DiffDock-L does not natively support multi-ligand SMILES string inputs, so in this work, we propose a modified inference procedure for DiffDock-L which *autoregressively* presents each (fragment) ligand SMILES string to the model while providing the same predicted protein structure to the model in each inference iteration (reporting for each complex the average confidence score over all iterations). Notably, as an inference-time modification, this sampling formulation permits multi-ligand sampling yet cannot model multi-ligand interactions directly and therefore often produces inter-ligand steric clashes.

b) As a single-ligand DL (flexible) docking method, **DynamicBind** adopts the same input and output formats as DiffDock-L with the following exceptions: (1) the predicted input protein structure is now flexible in response to (fragment) ligand docking; (2) the autoregressive inference procedure we adapted from that of DiffDock-L now provides DynamicBind with its own most recently predicted protein structure in each (fragment) ligand inference iteration, thereby providing the model with partial multi-ligand interaction context; and (3) iteration-averaged confidence scores *and* predicted affinities are reported for each complex. Nonetheless, for both DiffDock-L and DynamicBind, such modified inference procedures highlight the importance in future work of retraining such generative docking methods directly on multi-ligand complexes to address such inference-time compromises.

3. Formats for DL co-folding methods are as follows:

a) One of the first DL co-folding methods, **RoseTTAFold-All-Atom** is provided with a (multi-chain) protein sequence as well as (fragment) ligand

SMILES strings. The method is subsequently tasked with producing not only a (single) bound ligand conformation but also the bound protein conformation, using diverse MSA databases to provide evolutionary information to the model.

b) **NeuralPLexer** is a protein-ligand co-folding diffusion model trained using expansive PDB molecule and protein data sources. It receives as its inputs a (multi-chain) protein sequence as well as (fragment) ligand SMILES strings. The method is then tasked with producing multiple rank-ordered (flexible) protein-ligand structure conformations for each input complex, where we use the method's average ligand heavy atom plDDT scores for sampling ranking.

c) **AlphaFold 3** is a commercially-restricted biomolecular co-folding model trained on exhaustive PDB crystal structures and AlphaFold 2-predicted distillation structures. Following its default settings for inference, the model receives as its inputs a (multi-chain) protein sequence and (fragment) ligand SMILES strings, with default MSA and template inputs provided to the model. The method is then tasked with producing multiple rank-ordered (flexible) protein-ligand structure conformations for each input complex, using the method's intrinsic ranking score [27] for rank-ordering.

d) **Chai-1** is an open-source co-folding model (akin to AF3) trained on exhaustive PDB crystal structures and AlphaFold 2-predicted distillation structures along with AF3-based training protocols. Following its default settings for inference, the model receives as its inputs a (multi-chain) protein sequence and (fragment) ligand SMILES strings, with paired MSAs yet no template structures provided (as is its default setting). The method is then tasked with producing multiple rank-ordered protein-ligand bound

structure conformations for each input complex, using the method's intrinsic AF3-like ranking score for rank-ordering. Note that, as Chai-1's source code does not provide resources to generate multiple sequence alignments for input featurization, Chai-1 uses standardized (taxonomy-paired) multiple sequence alignments akin to those used by AF3 in all benchmarking experiments.

## E.7 ADDITIONAL RESULTS

In this section, we provide additional results for each baseline method using the Astex Diverse, PoseBusters Benchmark, and DockGen datasets as well as the CASP15 ligand prediction targets. Note that for all violin plots listed in this section, we curate them using combined results across each method's three independent runs (where applicable), in contrast to this section's bar charts where we instead report mean and standard deviation values across each method's three independent runs.

### E.7.1 Expanded primary ligand results

#### Primary ligand RMSD results

In Figures E.10, E.11, and E.12, we report the (binding site-superimposed) ligand RMSD values of each baseline method across the primary ligand Astex Diverse, PoseBusters Benchmark, and DockGen datasets, with molecular dynamics (MD)-based structural relaxation applied post-hoc. Overall, these figures demonstrate that AF3 and Chai-1 achieve the tightest RMSD distributions, except for single-sequence AF3 which occasionally produces catastrophic prediction errors by targeting incorrect PLI binding pockets. Further, these results show that MD-based relaxation generally does not modify the RMSD distribution of most baseline methods, except for DynamicBind and RFAA for which neither seem to benefit from such post-hoc optimizations.

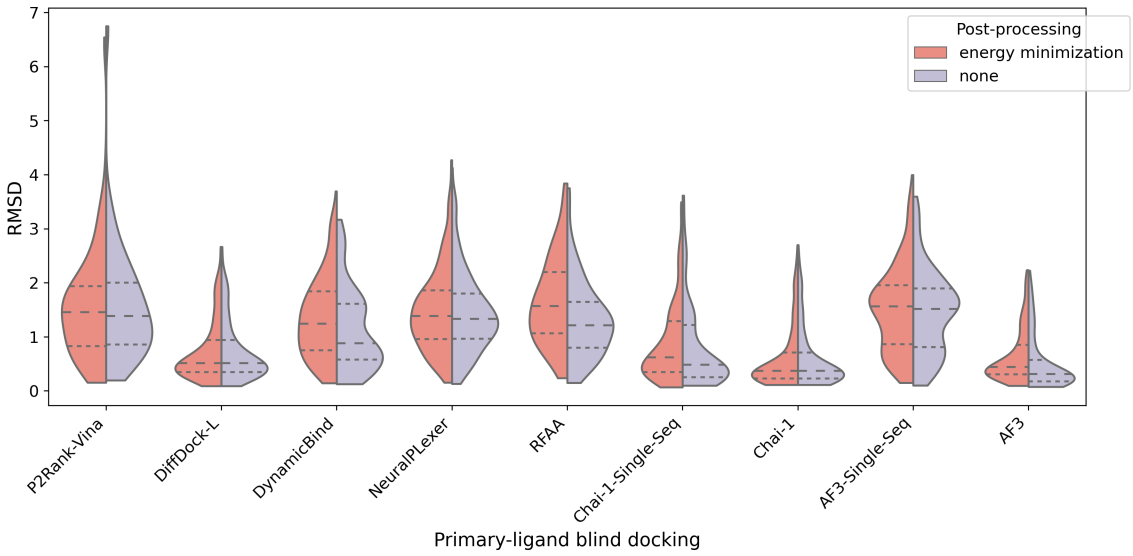



Figure E.10: Astex Diverse dataset results for primary ligand docking RMSD.

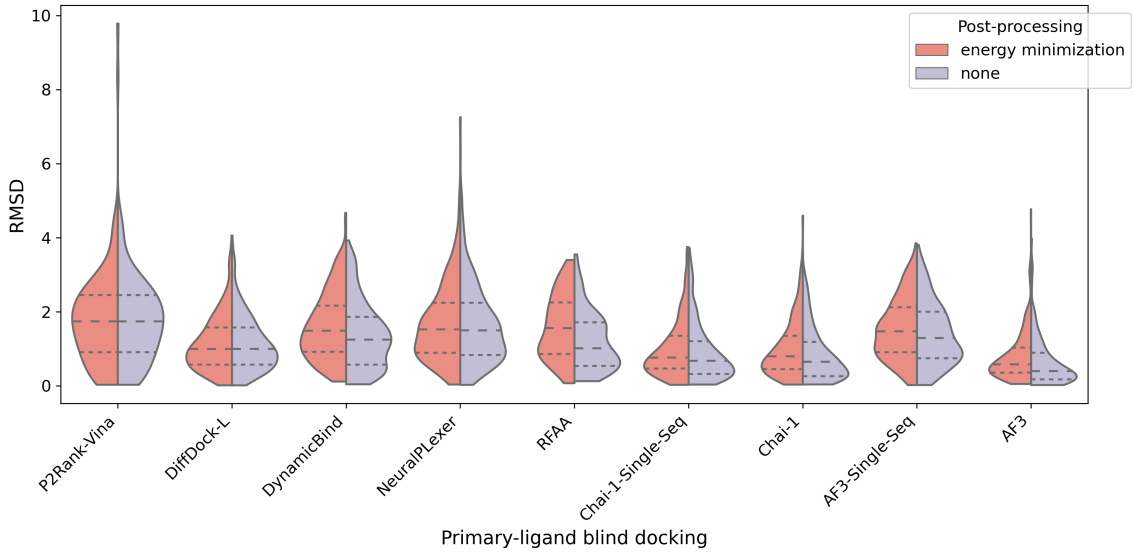



Figure E.11: PoseBusters Benchmark dataset results for primary ligand docking RMSD.

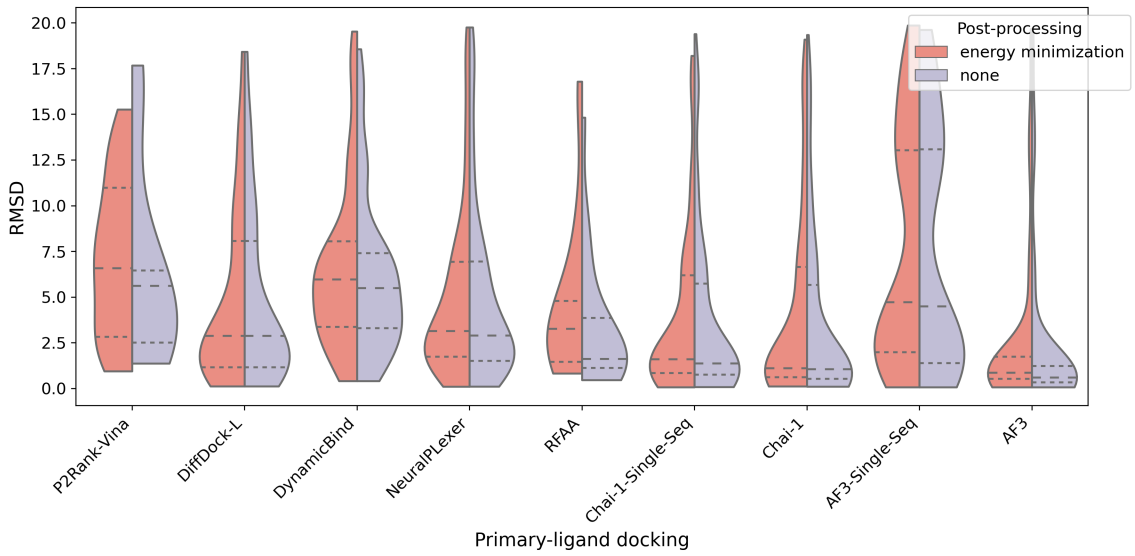



Figure E.12: DockGen dataset results for primary ligand docking RMSD.

### E.7.2 Expanded CASP15 results

#### Overview of expanded results

In this section, we begin by reporting additional CASP15 benchmarking results in terms of each baseline method's multi-ligand RMSD and lDDT-PLI distributions as violin plots. Subsequently, we report successful ligand docking success rates as well as RMSD and lDDT-PLI results specifically for the single-ligand (i.e., primary ligand) CASP15 targets. Lastly, we report all the above single and multi-ligand results specifically using only the CASP15 targets for which the crystal structures are publicly available, to facilitate reproducible future benchmarking.

#### Multi-ligand RMSD and lDDT-PLI

To start, Figures E.13, E.14, and E.15 report each method's multi-ligand RMSD and lDDT-PLI distributions as well as PB-Valid rates with and without relaxation. We see that AF3 produces the most tightly bound and accurate RMSD and lDDT-PLI distributions overall yet is challenged in its PB-Valid rate by the conventional method AutoDock Vina, highlighting that AF3 predicted several structurally accurate yet chemically implausible multi-ligand conformations for this dataset.

#### All single-ligand results

Next, Figures E.16, E.17, E.18, and E.19 display each method's single-ligand CASP15 docking success rates, PB-Valid rates, docking RMSD, and docking lDDT-PLI distributions, respectively. In summary, we can make a few respective observations from these plots. (1) AF3 achieves the highest structural accuracy for this subset of targets yet is challenged in its PLIF-WM rate by conventional and DL co-folding baseline methods such as AutoDock Vina and NeuralPLexer. (2) Even though most are positionally incorrect, structurally and chemically speaking, the majority of AutoDock

Vina and DiffDock-L's predictions are valid according to the PoseBusters software suite, whereas fewer of AF3's predictions are. (3) AutoDock Vina, DiffDock-L, NeuralPLexer, and AF3 yield notably lower RMSD distributions than all other baseline methods (including all single-sequence DL co-folding variants). (4) Only AF3 and AutoDock Vina produce a reasonable range of lDDT-PLI scores for these single-ligand targets.

### Single and multi-ligand results for *public* targets

Lastly, for completeness and reproducibility, Figures E.20, E.21, E.22, and E.23 present corresponding multi-ligand results for the public CASP15 targets, whereas Figures E.24, E.25, E.26, and E.27 report corresponding single-ligand results for the public CASP15 targets. Overall, we observe marginal differences between the full and public CASP15 target results for multi-ligand complexes, since once again AF3 achieves top results overall in the context of multi-ligands. However, we notice more striking performance drops between the full and public *single*-ligand CASP15 target results, suggesting that some of the private single-ligand complexes are easier prediction targets than most of the publicly available single-ligand complexes. In short, we find that AutoDock Vina consistently performs best in this single-ligand setting.

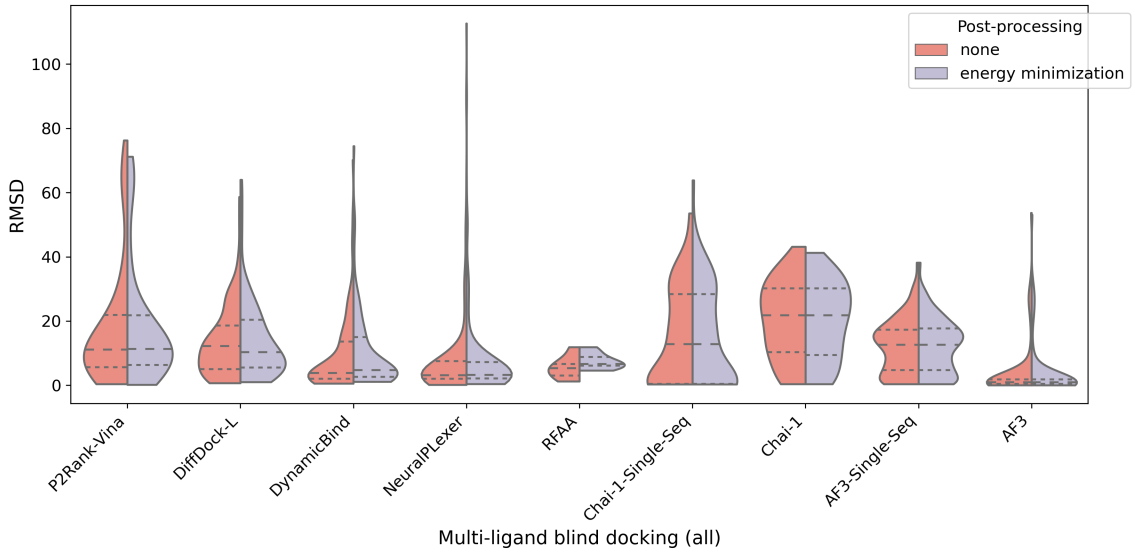



Figure E.13: CASP15 dataset results for multi-ligand docking RMSD with relaxation.

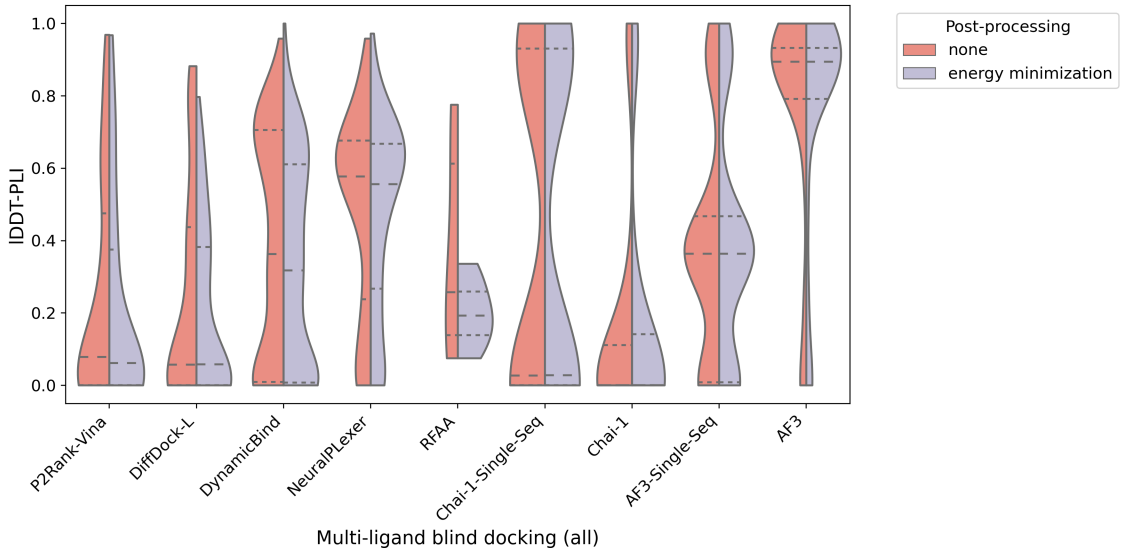



Figure E.14: CASP15 dataset results for multi-ligand docking lDDT-PLI with relaxation.

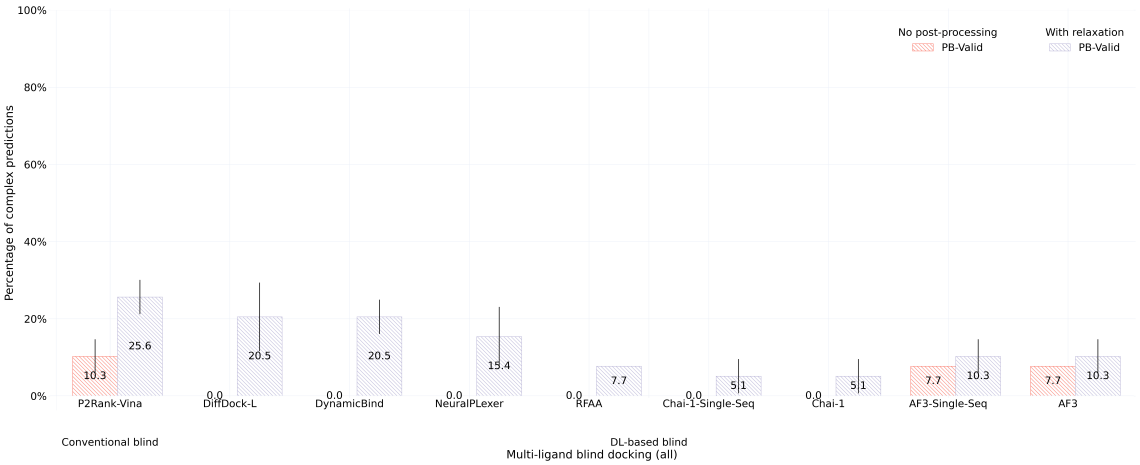



Figure E.15: CASP15 dataset results for multi-ligand docking PB-Valid rates with relaxation.

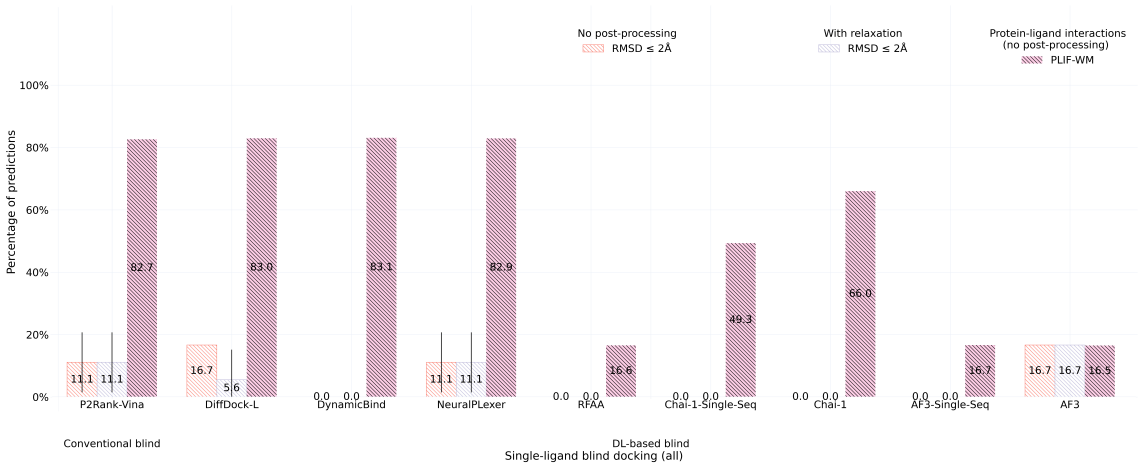



Figure E.16: CASP15 dataset results for successful single-ligand docking with relaxation.

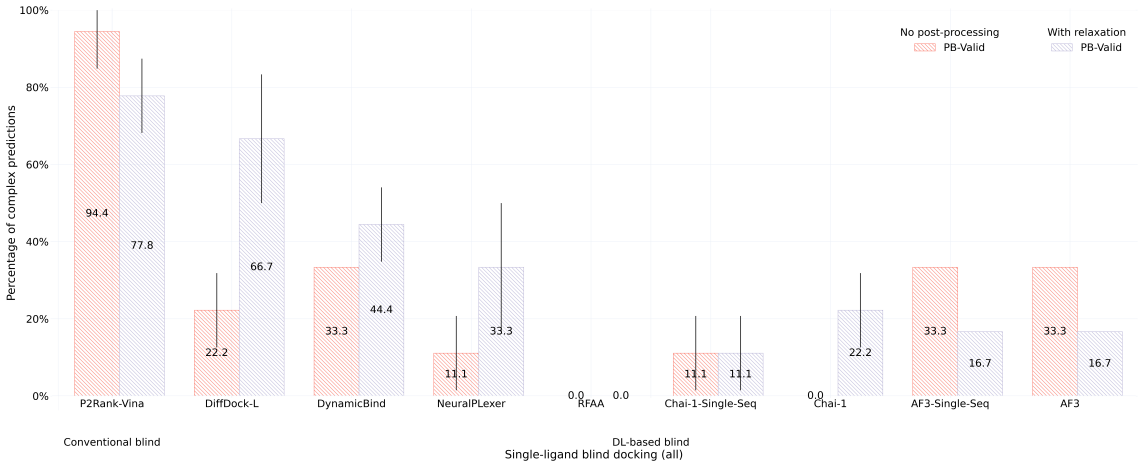



Figure E.17: CASP15 dataset results for single-ligand PB-Valid rates with relaxation.

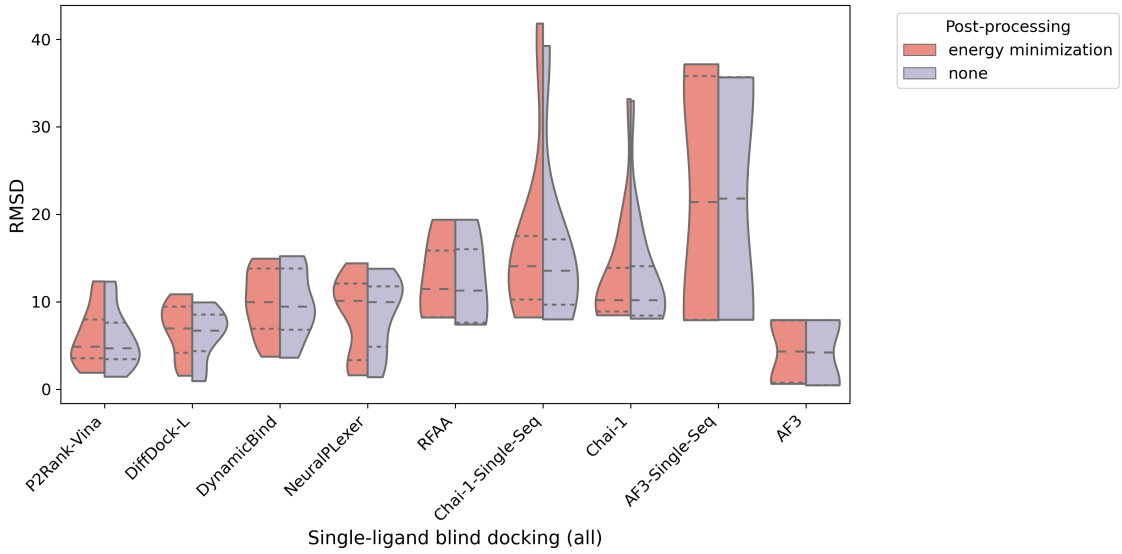



Figure E.18: CASP15 dataset results for single-ligand docking RMSD with relaxation.

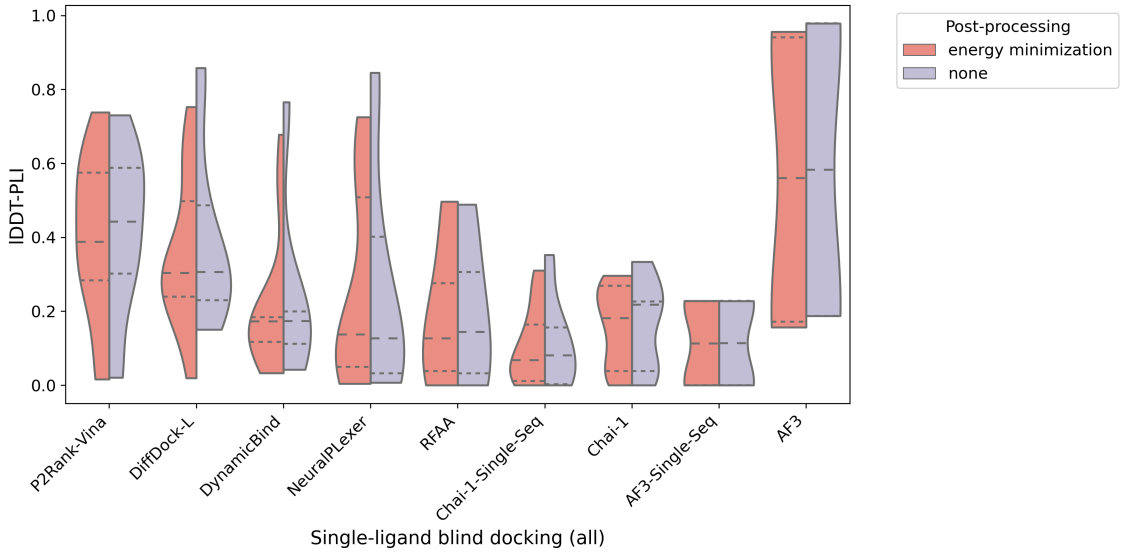



Figure E.19: CASP15 dataset results for single-ligand docking lDDT-PLI with relaxation.

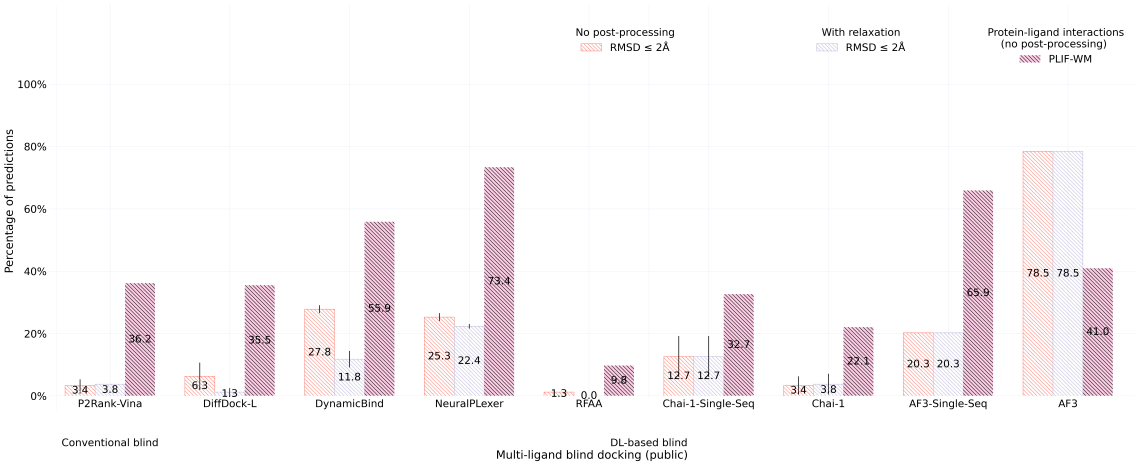



Figure E.20: CASP15 public dataset results for successful multi-ligand docking with relaxation.

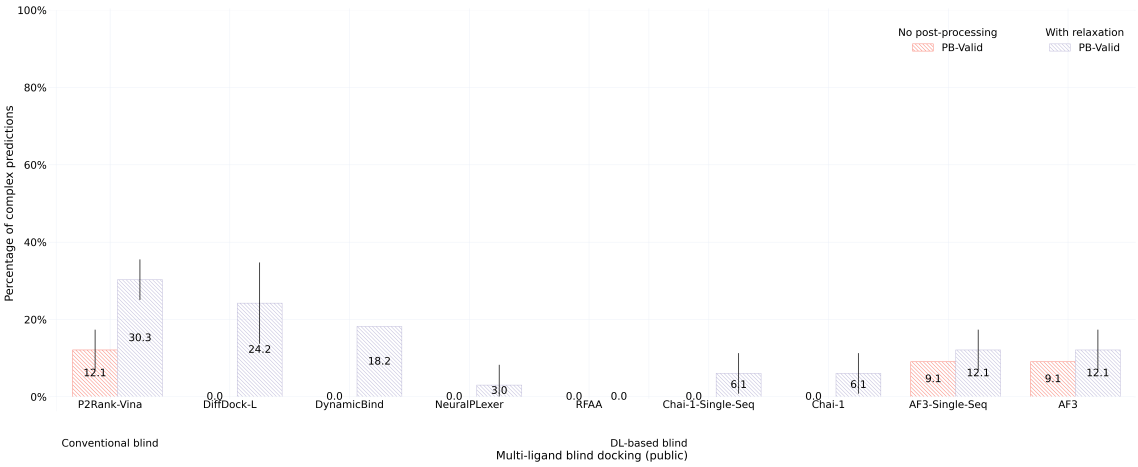



Figure E.21: CASP15 public dataset results for multi-ligand PB-Valid rates with relaxation.

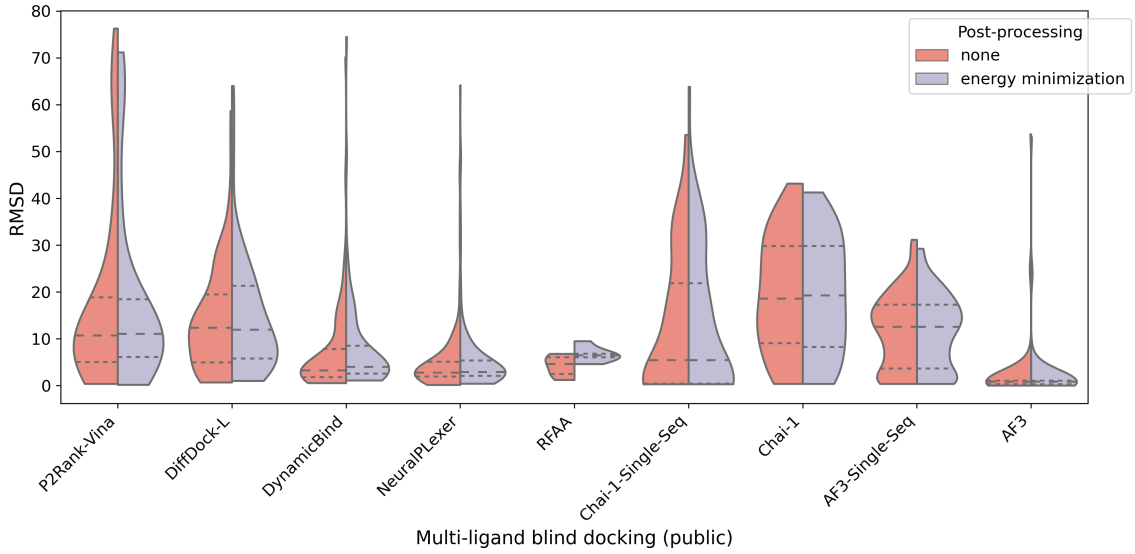



Figure E.22: CASP15 public dataset results for multi-ligand docking RMSD with relaxation.

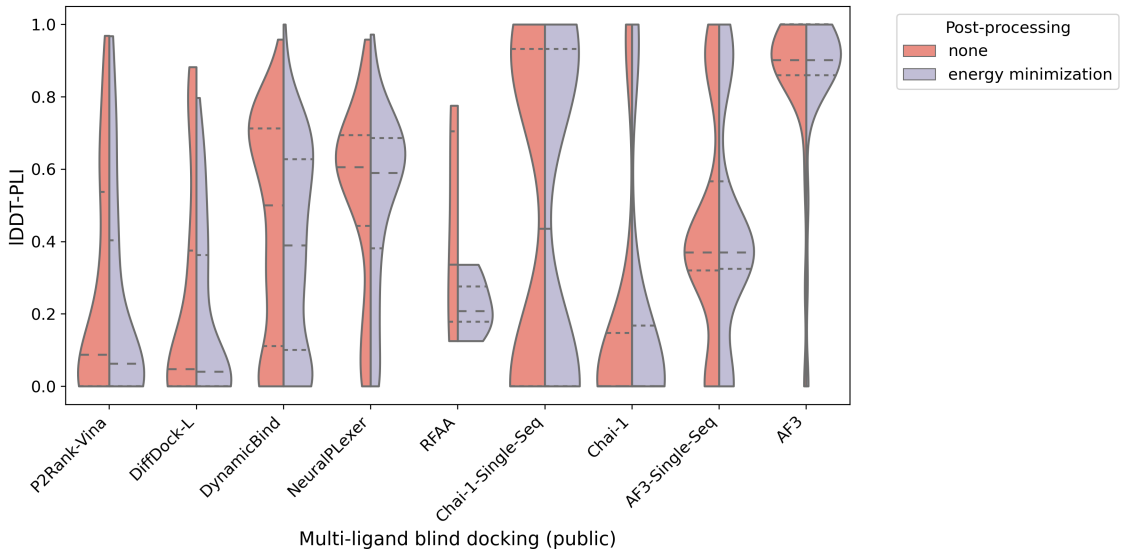



Figure E.23: CASP15 public dataset results for multi-ligand docking lDDT-PLI with relaxation.

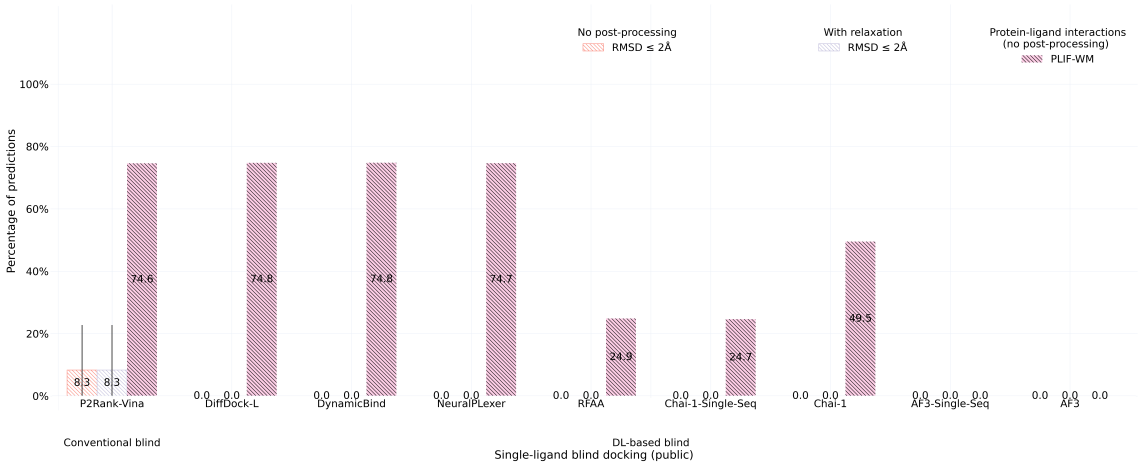



Figure E.24: CASP15 public dataset results for successful single-ligand docking with relaxation.

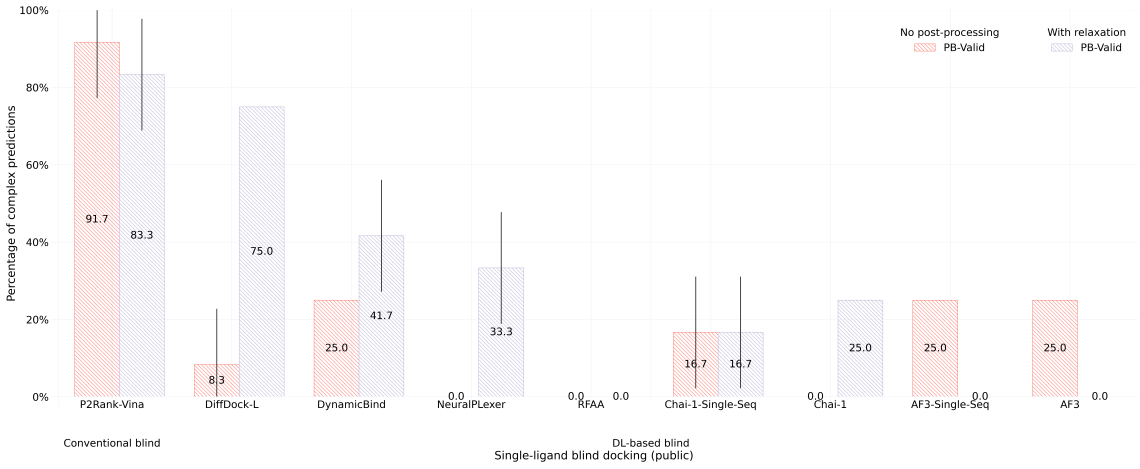



Figure E.25: CASP15 public dataset results for single-ligand PB-Valid rates with relaxation.

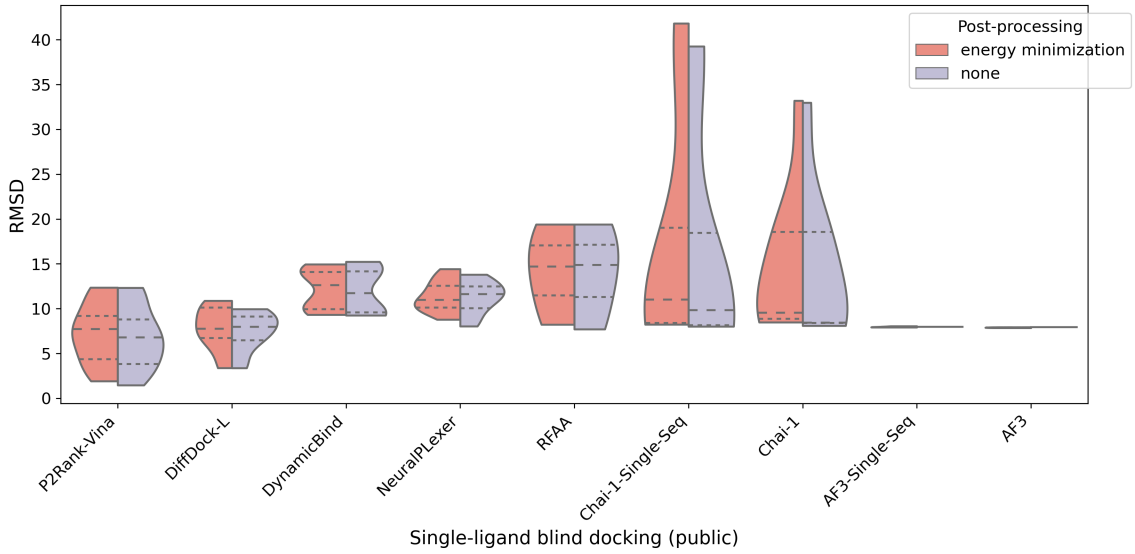



Figure E.26: CASP15 public dataset results for single-ligand docking RMSD with relaxation.

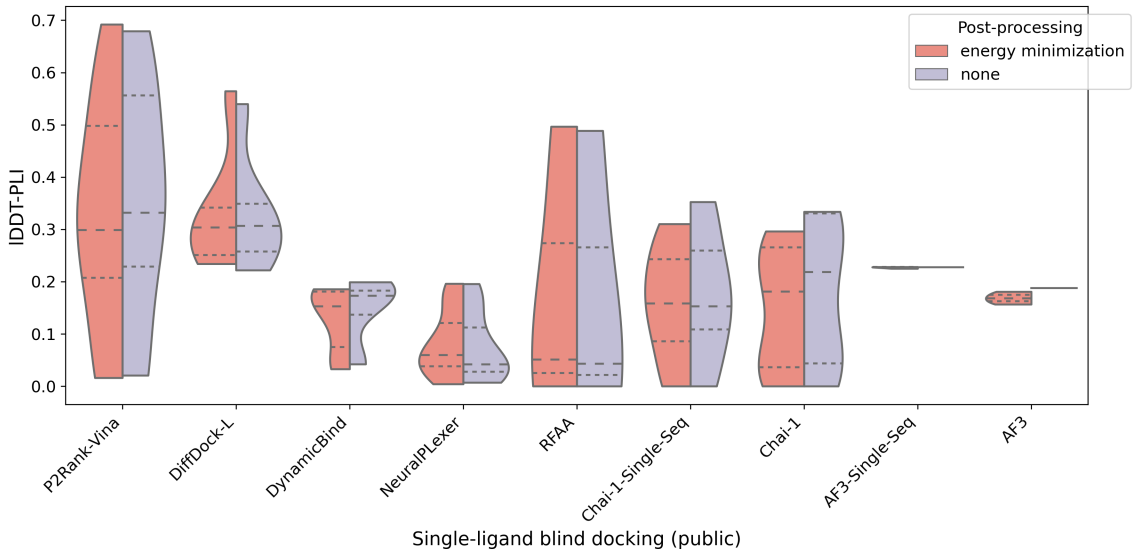



Figure E.27: CASP15 public dataset results for single-ligand docking lDDT-PLI with relaxation.

# BIBLIOGRAPHY


[1] M. J. Berridge. "The molecular basis of communication within the cell". In: *Scientific American* 253.4 (1985), 142–152A.

[2] A. C. Wilson. "The molecular basis of evolution". In: *Scientific American* 253.4 (1985), pp. 164–175.

[3] F. Crick. "Central dogma of molecular biology". In: *Nature* 227.5258 (1970), pp. 561–563.

[4] J. A. Farías-Rico and C. M. Mourra-Díaz. "A short tale of the origin of proteins and ribosome evolution". In: *Microorganisms* 10.11 (2022), p. 2115.

[5] R. B. Gennis. *Biomembranes: molecular structure and function*. Springer Science & Business Media, 2013.

[6] A. Ilari and C. Savino. "Protein structure determination by x-ray crystallography". In: *Bioinformatics: Data, Sequence Analysis and Evolution* (2008), pp. 63–87.

[7] K. Wüthrich. "Protein structure determination in solution by NMR spectroscopy." In: *Journal of Biological Chemistry* 265.36 (1990), pp. 22059–22062.

[8] K. M. Yip, N. Fischer, E. Paknia, A. Chari, and H. Stark. "Atomic-resolution protein structure determination by cryo-EM". In: *Nature* 587.7832 (2020), pp. 157–161.

[9] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[10] P. J. Werbos. “Backpropagation through time: what it does and how to do it”. In: *Proceedings of the IEEE* 78.10 (1990), pp. 1550–1560.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[12] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. “The graph neural network model”. In: *IEEE transactions on neural networks* 20.1 (2008), pp. 61–80.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012).

[14] H. Wang, T. Fu, Y. Du, W. Gao, K. Huang, Z. Liu, P. Chandak, S. Liu, P. Van Katwyk, A. Deac, et al. “Scientific discovery in the age of artificial intelligence”. In: *Nature* 620.7972 (2023), pp. 47–60.

[15] 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”. In: *nature* 596.7873 (2021), pp. 583–589.

[16] K. Atz, F. Grisoni, and G. Schneider. “Geometric deep learning on molecular representations”. In: *Nature Machine Intelligence* 3.12 (2021), pp. 1023–1032.

[17] B. Jing, S. Eismann, P. Suriana, R. J. L. Townshend, and R. Dror. “Learning from Protein Structure with Geometric Vector Perceptrons”. In: *International Conference on Learning Representations*. 2021.

[18] V. G. Satorras, E. Hoogeboom, and M. Welling. “E(n) equivariant graph neural networks”. In: *International conference on machine learning*. PMLR. 2021, pp. 9323–9332.

[19] 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". In: *Science* 373.6557 (2021), pp. 871–876.

[20] R. J. Townshend, S. Eismann, A. M. Watkins, R. Rangan, M. Karelina, R. Das, and R. O. Dror. "Geometric deep learning of RNA structure". In: *Science* 373.6558 (2021), pp. 1047–1051.

[21] M. Geiger and T. Smidt. "e3nn: Euclidean neural networks". 2022.

[22] Z. Zhang, M. Xu, A. Jamasb, V. Vijil, A. Lozano, P. Das, and J. Tang. "Protein Representation Learning by Geometric Structure Pretraining". In: *International Conference on Machine Learning*. 2022.

[23] C. Hsu, R. Verkuil, J. Liu, Z. Lin, B. Hie, T. Sercu, A. Lerer, and A. Rives. "Learning inverse folding from millions of predicted structures". In: *International conference on machine learning*. PMLR. 2022, pp. 8946–8970.

[24] G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. S. Jaakkola. "DiffDock: Diffusion Steps, Twists, and Turns for Molecular Docking". In: *International Conference on Learning Representations*. 2023.

[25] 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". In: *Science* 379.6637 (2023), pp. 1123–1130.

[26] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges". 2021.

[27] J. Abramson, J. Adler, J. Dunger, R. Evans, T. Green, A. Pritzel, O. Ronneberger, L. Willmore, A. J. Ballard, J. Bambrick, et al. "Accurate structure

prediction of biomolecular interactions with AlphaFold 3". In: *Nature* 630.8016 (2024), pp. 493–500.

[28] W. Lu, J. Zhang, W. Huang, Z. Zhang, X. Jia, Z. Wang, L. Shi, C. Li, P. G. Wolynes, and S. Zheng. "DynamicBind: predicting ligand-specific protein-ligand complex structure with a deep equivariant generative model". In: *Nature Communications* 15.1 (2024), p. 1071.

[29] Z. Qiao, W. Nie, A. Vahdat, T. F. Miller III, and A. Anandkumar. "State-specific protein–ligand complex structure prediction with a multiscale deep generative model". In: *Nature Machine Intelligence* 6.2 (2024), pp. 195–208.

[30] R. Krishna, J. Wang, W. Ahern, P. Sturmfels, P. Venkatesh, I. Kalvet, G. R. Lee, F. S. Morey-Burrows, I. Anishchenko, I. R. Humphreys, et al. "Generalized biomolecular modeling and design with RoseTTAFold All-Atom". In: *Science* 384.6693 (2024), eadl2528.

[31] P. Bryant, A. Kelkar, A. Guljas, C. Clementi, and F. Noé. "Structure prediction of protein-ligand complexes from sequence information with Umol". In: *Nature Communications* 15.1 (2024), p. 4536.

[32] M. Baek, R. McHugh, I. Anishchenko, H. Jiang, D. Baker, and F. DiMaio. "Accurate prediction of protein–nucleic acid complexes using RoseTTAFoldNA". In: *Nature methods* 21.1 (2024), pp. 117–121.

[33] J. A. Wells and C. L. McClendon. "Reaching for high-hanging fruit in drug discovery at protein–protein interfaces". In: *Nature* 450.7172 (2007), pp. 1001–1009.

[34] Y. Murakami, L. P. Tripathi, P. Prathipati, and K. Mizuguchi. "Network analysis and in silico prediction of protein–protein interactions with applications in drug discovery". In: *Current opinion in structural biology* 44 (2017), pp. 134–142.

[35] Y. Liu, H. Yuan, L. Cai, and S. Ji. "Deep learning of high-order interactions for protein interface prediction". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 679–687.

[36] C. Chen, T. Wu, Z. Guo, and J. Cheng. "Combination of deep neural network with attention mechanism enhances the explainability of protein contact prediction". In: *Proteins: Structure, Function, and Bioinformatics* 89.6 (2021), pp. 697–707. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.26052. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.26052.

[37] A. Morehead, C. Chen, A. Sedova, and J. Cheng. "DIPS-Plus: The enhanced database of interacting protein structures for interface prediction". In: *Scientific Data* 10.1 (2023), p. 509.

[38] D. Sehnal, S. Bittrich, M. Deshpande, R. Svobodová, K. Berka, V. Bazgier, S. Velankar, S. K. Burley, J. Koča, and A. S. Rose. "Mol* Viewer: modern web app for 3D visualization and analysis of large biomolecular structures". In: *Nucleic Acids Research* 49.W1 (May 2021), W431–W437. ISSN: 0305-1048. eprint: https://academic.oup.com/nar/article-pdf/49/W1/W431/38842088/gkab314.pdf. URL: https://doi.org/10.1093/nar/gkab314.

[39] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko. "Quantum-chemical insights from deep tensor neural networks". In: *Nature communications* 8.1 (2017), pp. 1–8.

[40] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur. "Protein Interface Prediction using Graph Convolutional Networks". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc.,

2017, pp. 6530–6539. URL: https://proceedings.neurips.cc/paper/2017/file/f507783927f2ec2737ba40afbd17efb5-Paper.pdf.

[41] T. Vreven, I. H. Moal, A. Vangone, B. G. Pierce, P. L. Kastritis, M. Torchala, R. Chaleil, B. Jiménez-García, P. A. Bates, J. Fernandez-Recio, et al. "Updates to the integrated protein–protein interaction benchmarks: docking benchmark version 5 and affinity benchmark version 2". In: *Journal of molecular biology* 427.19 (2015), pp. 3031–3041.

[42] R. Townshend, R. Bedi, P. Suriana, and R. Dror. "End-to-End Learning on 3D Protein Structure for Interface Prediction". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019, pp. 15642–15651. URL: https://proceedings.neurips.cc/paper/2019/file/6c7de1f27f7de61a6daddfffbe05c058-Paper.pdf.

[43] 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". In: *Nature Methods* 17.2 (2020), pp. 184–192.

[44] B. Dai and C. Bailey-Kellogg. "Protein Interaction Interface Region Prediction by Geometric Deep Learning". In: *Bioinformatics* (Mar. 2021). btab154. ISSN: 1367-4803. eprint: https://academic.oup.com/bioinformatics/advance-article-pdf/doi/10.1093/bioinformatics/btab154/36516110/btab154.pdf. URL: https://doi.org/10.1093/bioinformatics/btab154.

[45] S. Ahmad and K. Mizuguchi. "Partner-aware prediction of interacting residues in protein-protein complexes from sequence data". In: *PloS one* 6.12 (2011), e29104.

[46] X. Liu, Y. Luo, P. Li, S. Song, and J. Peng. "Deep geometric representations for modeling effects of mutations on protein-protein binding affinity". In: *PLoS computational biology* 17.8 (2021), e1009284.

[47] A. Costa, P. Chatterjee, M. Ponnapati, S. Bhat, K. Palepu, J. Jacobson, and I. Drori. "End-to-end Euclidean equivariant Transformers for protein docking". In: *NeurIPS Workshop on Learning Meaningful Representations of Life* (2021).

[48] M. Steinegger, M. Meier, M. Mirdita, H. Vöhringer, S. J. Haunsberger, and J. Söding. "HH-suite3 for fast remote homology detection and deep protein annotation". In: *BMC bioinformatics* 20.1 (2019), pp. 1–15.

[49] M. F. Lensink, G. Brysbaert, N. Nadzirin, S. Velankar, R. A. Chaleil, T. Gerguri, P. A. Bates, E. Laine, A. Carbone, S. Grudinin, et al. "Blind prediction of homo-and hetero-protein complexes: The CASP13-CAPRI experiment". In: *Proteins: Structure, Function, and Bioinformatics* 87.12 (2019), pp. 1200–1221.

[50] M. F. Lensink, G. Brysbaert, T. Mauri, N. Nadzirin, S. Velankar, R. A. Chaleil, T. Clarence, P. A. Bates, R. Kong, B. Liu, et al. "Prediction of protein assemblies, the next frontier: The CASP14-CAPRI experiment". In: *Proteins: Structure, Function, and Bioinformatics* (2021).

[51] R. A. Jordan, E.-M. Yasser, D. Dobbs, and V. Honavar. "Predicting protein-protein interface residues using local surface structural similarity". In: *BMC bioinformatics* 13.1 (2012), pp. 1–14.

[52] J. Yang, A. Roy, and Y. Zhang. "Protein–ligand binding site recognition using complementary binding-specific substructure comparison and sequence profile alignment". In: *Bioinformatics* 29.20 (2013), pp. 2588–2595.

[53] J. Ingraham, V. Garg, R. Barzilay, and T. Jaakkola. "Generative Models for Graph-Based Protein Design". In: *Advances in Neural Information Processing*

*Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/file/f3a4ff4839c56a5f460c88cce3666a2b-Paper.pdf.

[54] V. P. Dwivedi and X. Bresson. "A Generalization of Transformer Networks to Graphs". In: *DLG-AAAI 2021 Workshop*. 2021.

[55] M. S. Hussain, M. J. Zaki, and D. Subramanian. "Global Self-Attention as a Replacement for Graph Convolution". In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD '22. Washington DC, USA: Association for Computing Machinery, 2022, pp. 655–665. ISBN: 9781450393850. URL: https://doi.org/10.1145/3534678.3539296.

[56] J. Hu, L. Shen, and G. Sun. "Squeeze-and-excitation networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 7132–7141.

[57] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *ICLR (Poster)*. 2015. URL: http://arxiv.org/abs/1412.6980.

[58] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. "Averaging weights leads to wider optima and better generalization". In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Association For Uncertainty in Artificial Intelligence (AUAI). 2018, pp. 876–885.

[59] R. Sanchez-Garcia, C. O. S. Sorzano, J. M. Carazo, and J. Segura. "BIPSPI: a method for the prediction of partner-specific protein–protein interfaces". In: *Bioinformatics* 35.3 (July 2018), pp. 470–477. ISSN: 1367-4803. eprint: https://academic.oup.com/bioinformatics/article-pdf/35/3/470/27700304/bty647.pdf. URL: https://doi.org/10.1093/bioinformatics/bty647.

[60] Y. Yan and S.-Y. Huang. "Accurate prediction of inter-protein residue–residue contacts for homo-oligomeric protein complexes". In: *Briefings in Bioinformatics* (2021).

[61] 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". In: *Nucleic acids research* 46.W1 (2018), W432–W437.

[62] T. N. Kipf and M. Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=SJU4ayYgl.

[63] J.-Y. Jiang, P. H. Chen, C.-J. Hsieh, and W. Wang. "Clustering and constructing user coresets to accelerate large-scale top-k recommender systems". In: *Proceedings of The Web Conference 2020*. 2020, pp. 2177–2187.

[64] X. Chen*, A. Morehead*, J. Liu, and J. Cheng. "A gated graph transformer for protein complex structure quality assessment and its performance in CASP15". In: *Intelligent Systems for Molecular Biology (ISMB)*. 2023.

[65] H. Ma, Y. Bian, Y. Rong, W. Huang, T. Xu, W. Xie, G. Ye, and J. Huang. "Cross-dependent graph neural networks for molecular property prediction". In: *Bioinformatics* 38.7 (2022), pp. 2003–2009.

[66] Y. Wu, M. Gao, M. Zeng, J. Zhang, and M. Li. "BridgeDPI: a novel graph neural network for predicting drug–protein interactions". In: *Bioinformatics* 38.9 (2022), pp. 2571–2578.

[67] M. Karimi, D. Wu, Z. Wang, and Y. Shen. "DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks". In: *Bioinformatics* 35.18 (2019), pp. 3329–3338.

[68] F. Baldassarre, D. Menéndez Hurtado, A. Elofsson, and H. Azizpour. "GraphQA: protein model quality assessment using graph convolutional networks". In: *Bioinformatics* 37.3 (2021), pp. 360–366.

[69] T. Xia and W.-S. Ku. "Geometric graph representation learning on protein structure prediction". In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021, pp. 1873–1883.

[70] A. Morehead, X. Chen, T. Wu, J. Liu, and J. Cheng. "EGR: Equivariant Graph Refinement and Assessment of 3D Protein Complex Structures". 2022.

[71] K. Wang, R. Zhou, J. Tang, and M. Li. "GraphscoreDTA: optimized graph neural network for protein–ligand binding affinity prediction". In: *Bioinformatics* 39.6 (May 2023), btad340. ISSN: 1367-4811.

[72] A. Morehead, C. Chen, and J. Cheng. "Geometric Transformers for Protein Interface Contact Prediction". In: *The Tenth International Conference on Learning Representations (ICLR)*. 2022.

[73] T. Cohen and M. Welling. "Group equivariant convolutional networks". In: *International conference on machine learning*. PMLR. 2016, pp. 2990–2999.

[74] N. Thomas, T. E. Smidt, S. M. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. F. Riley. "Tensor Field Networks: Rotation- and Translation-Equivariant Neural Networks for 3D Point Clouds". 2018.

[75] S. Batzner, A. Musaelian, L. Sun, et al. "E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials". In: *Nature communications* 13.1 (2022), p. 2453.

[76] B. Jing, S. Eismann, P. N. Soni, and R. O. Dror. "Equivariant graph neural networks for 3d macromolecular structure". In: *ICML 2021 Workshop on Computational Biology*. 2021.

[77] W. Du, H. Zhang, Y. Du, Q. Meng, W. Chen, N. Zheng, B. Shao, and T.-Y. Liu. "SE (3) equivariant graph neural networks with complete local frames". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 5583–5608.

[78] F. B. Fuchs, E. Wagstaff, J. Dauparas, and I. Posner. "Iterative se (3)-transformers". In: *Geometric Science of Information: 5th International Conference, GSI 2021, Paris, France, July 21–23, 2021, Proceedings 5*. Springer. 2021, pp. 585–595.

[79] L. Wang, H. Liu, Y. Liu, J. Kurtin, and S. Ji. "Learning Hierarchical Protein Representations via Complete 3D Graph Networks". In: *International Conference on Learning Representations*. 2023.

[80] L. Wang, Y. Liu, Y. Lin, H. Liu, and S. Ji. "ComENet: Towards Complete and Efficient Message Passing for 3D Molecular Graphs". In: *Advances in Neural Information Processing Systems*. Ed. by A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho. 2022.

[81] K. Adams, L. Pattanaik, and C. W. Coley. "Learning 3D Representations of Molecular Chirality with Invariance to Bond Rotations". In: *International Conference on Learning Representations*. 2022. URL: `https://openreview.net/forum?id=hm2tNDdgaFK`.

[82] E. E. Bolton, J. Chen, S. Kim, L. Han, S. He, W. Shi, V. Simonyan, Y. Sun, P. A. Thiessen, J. Wang, et al. "PubChem3D: a new resource for scientists". In: *Journal of cheminformatics* 3 (2011), pp. 1–15.

[83] A. Schneuing, C. Harris, Y. Du, K. Didi, A. Jamasb, I. Igashov, W. Du, C. Gomes, T. L. Blundell, P. Lio, et al. "Structure-based drug design with equivariant diffusion models". In: *Nature Computational Science* 4.12 (2024), pp. 899–909.

[84] R. J. L. Townshend, M. Vögele, P. A. Suriana, A. Derry, A. Powers, Y. Laloudakis, S. Balachandar, B. Jing, B. M. Anderson, S. Eismann, et al. "ATOM3D: Tasks on Molecules in Three Dimensions". In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. 2021.

[85] M. A. Rezaei, Y. Li, D. Wu, X. Li, and C. Li. "Deep learning in drug design: protein-ligand binding affinity prediction". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2020).

[86] S. Aykent and T. Xia. "GBPNet: Universal Geometric Representation Learning on Protein Structures". In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD '22. Washington DC, USA: Association for Computing Machinery, 2022, pp. 4–14. ISBN: 9781450393850.

[87] S. Liu, H. Guo, and J. Tang. "Molecular Geometry Pretraining with SE(3)-Invariant Denoising Distance Matching". In: *International Conference on Learning Representations*. 2023.

[88] M. Fey and J. E. Lenssen. "Fast Graph Representation Learning with PyTorch Geometric". In: *ICLR 2019 RLGM Workshop*. 2019.

[89] K. Schütt, O. Unke, and M. Gastegger. "Equivariant message passing for the prediction of tensorial properties and molecular spectra". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 9377–9388.

[90] P. Thölke and G. De Fabritiis. "Equivariant transformers for neural network based molecular potentials". In: *International Conference on Learning Representations*. 2022.

[91] R. Wang, X. Fang, Y. Lu, C.-Y. Yang, and S. Wang. "The PDBbind database: methodologies and updates". In: *Journal of medicinal chemistry* 48.12 (2005), pp. 4111–4119.

[92] A. Zemla. “LGA: a method for finding 3D similarities in protein structures”. In: *Nucleic acids research* 31.13 (2003), pp. 3370–3374.

[93] A. Kryshtafovych, T. Schwede, M. Topf, K. Fidelis, and J. Moult. “Critical assessment of methods of protein structure prediction (CASP)—Round XIV”. In: *Proteins: Structure, Function, and Bioinformatics* 89.12 (2021), pp. 1607–1617.

[94] A. B. Dieng, Y. Kim, A. M. Rush, and D. M. Blei. “Avoiding latent variable collapse with generative skip models”. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR. 2019, pp. 2397–2405.

[95] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.

[96] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro. “Diffwave: A versatile diffusion model for audio synthesis”. In: *arXiv preprint arXiv:2009.09761* (2020).

[97] W. Peebles, I. Radosavovic, T. Brooks, A. A. Efros, and J. Malik. “Learning to learn with generative models of neural network checkpoints”. In: *arXiv preprint arXiv:2209.12892* (2022).

[98] N. Anand and T. Achim. “Protein structure and sequence generation with equivariant denoising diffusion probabilistic models”. In: *arXiv preprint arXiv:2205.15019* (2022).

[99] G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. Jaakkola. “Diffdock: Diffusion steps, twists, and turns for molecular docking”. In: *arXiv preprint arXiv:2210.01776* (2022).

[100] Z. Guo, J. Liu, Y. Wang, M. Chen, Wang, D. D. Xu, and J. Cheng. "Diffusion models in bioinformatics and computational biology". In: *Nature Reviews Bioengineering* (2023).

[101] J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, et al. "De novo design of protein structure and function with RFdiffusion". In: *Nature* 620.7976 (2023), pp. 1089–1100.

[102] A. Morehead, A. Bhatnagar, J. A. Ruffolo, and A. Madani. "Towards Joint Sequence-Structure Generation of Nucleic Acid and Protein Complexes". In: *NeurIPS Machine Learning in Structural Biology (MLSB) Workshop*. 2023.

[103] M. Xu, L. Yu, Y. Song, C. Shi, S. Ermon, and J. Tang. "Geodiff: A geometric diffusion model for molecular conformation generation". In: *arXiv preprint arXiv:2203.02923* (2022).

[104] N. W. Gebauer, M. Gastegger, S. S. Hessmann, K.-R. Müller, and K. T. Schütt. "Inverse design of 3d molecular structures with conditional generative neural networks". In: *Nature communications* 13.1 (2022), p. 973.

[105] D. M. Anstine and O. Isayev. "Generative Models as an Emerging Paradigm in the Chemical Sciences". In: *Journal of the American Chemical Society* 145.16 (2023), pp. 8736–8750.

[106] N. Mudur and D. P. Finkbeiner. "Can denoising diffusion probabilistic models generate realistic astrophysical fields?" In: *arXiv preprint arXiv:2211.12444* (2022).

[107] C. K. Joshi, C. Bodnar, S. V. Mathis, T. Cohen, and P. Lio. "On the expressive power of geometric graph neural networks". In: *International conference on machine learning*. PMLR. 2023, pp. 15330–15355.

[108] H. Stärk, O. Ganea, L. Pattanaik, R. Barzilay, and T. Jaakkola. "Equibind: Geometric deep learning for drug binding structure prediction". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 20503–20521.

[109] A. R. Jamasb*, A. Morehead*, Z. Zhang*, C. K. Joshi*, K. Didi, S. V. Mathis, C. Harris, J. Tang, J. Cheng, P. Lio, and T. L. Blundell. "Evaluating Representation Learning on the Protein Structure Universe". In: *The Twelfth International Conference on Learning Representations (ICLR)*. Also presented at the NeurIPS 2023 MLSB workshop. 2024.

[110] A. Morehead, J. Liu, and J. Cheng. "Protein Structure Accuracy Estimation using Geometry-Complete Perceptron Networks". In: *Protein Science* (2024).

[111] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. "Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds". In: *arXiv preprint arXiv:1802.08219* (2018).

[112] M. Buttenschoen, G. M. Morris, and C. M. Deane. "PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences". In: *Chemical Science* (2024).

[113] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. Von Lilienfeld. "Quantum chemistry structures and properties of 134 kilo molecules". In: *Scientific data* 1.1 (2014), pp. 1–7.

[114] E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling. "Equivariant diffusion for molecule generation in 3d". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 8867–8887.

[115] B. Anderson, T. S. Hy, and R. Kondor. "Cormorant: Covariant molecular neural networks". In: *Advances in neural information processing systems* 32 (2019).

[116] V. G. Satorras, E. Hoogeboom, F. B. Fuchs, I. Posner, and M. Welling. "E (n) equivariant normalizing flows". In: *arXiv preprint arXiv:2105.09016* (2021).

[117] G. Landrum et al. "RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling". In: *Greg Landrum* 8 (2013).

[118] R. Krishna, J. Wang, W. Ahern, P. Sturmfels, P. Venkatesh, I. Kalvet, G. R. Lee, F. S. Morey-Burrows, I. Anishchenko, I. R. Humphreys, et al. "Generalized Biomolecular Modeling and Design with RoseTTAFold All-Atom". In: *bioRxiv* (2023), pp. 2023–10.

[119] DeepMind-Isomorphic. "Performance and structural coverage of the latest, in-development AlphaFold model". In: *DeepMind* (2023).

[120] N. Gebauer, M. Gastegger, and K. Schütt. "Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules". In: *Advances in neural information processing systems* 32 (2019).

[121] L. Wu, C. Gong, X. Liu, M. Ye, and Q. Liu. "Diffusion-based molecule generation with informative prior bridges". In: *arXiv preprint arXiv:2209.00865* (2022).

[122] M. Xu, A. Powers, R. Dror, S. Ermon, and J. Leskovec. "Geometric Latent Diffusion Models for 3D Molecule Generation". In: *arXiv preprint arXiv:2305.01140* (2023).

[123] C. Vignac, N. Osman, L. Toni, and P. Frossard. "Midi: Mixed graph and 3d denoising diffusion for molecule generation". In: *arXiv preprint arXiv:2302.09048* (2023).

[124] T. Le, J. Cremer, F. Noé, D.-A. Clevert, and K. Schütt. "Navigating the Design Space of Equivariant Diffusion-Based Generative Models for De Novo 3D Molecule Generation". In: *arXiv preprint arXiv:2309.17296* (2023).

[125] D. G. Smith, L. A. Burns, A. C. Simmonett, R. M. Parrish, M. C. Schieber, R. Galvelis, P. Kraus, H. Kruse, R. Di Remigio, A. Alenaizan, et al. "PSI4 1.4: Open-source software for high-throughput quantum chemistry". In: *The Journal of chemical physics* 152.18 (2020).

[126] S. Lehtola, C. Steigemann, M. J. Oliveira, and M. A. Marques. "Recent developments in libxc—A comprehensive library of functionals for density functional theory". In: *SoftwareX* 7 (2018), pp. 1–5.

[127] P. Pracht, F. Bohle, and S. Grimme. "Automated exploration of the low-energy chemical space with fast quantum chemical methods". In: *Physical Chemistry Chemical Physics* 22.14 (2020), pp. 7169–7192.

[128] S. Axelrod and R. Gomez-Bombarelli. "GEOM, energy-annotated molecular conformations for property prediction and molecular generation". In: *Scientific Data* 9.1 (2022), p. 185.

[129] A. K. Rappé, C. J. Casewit, K. Colwell, W. A. Goddard III, and W. M. Skiff. "UFF, a full periodic table force field for molecular mechanics and molecular dynamics simulations". In: *Journal of the American chemical society* 114.25 (1992), pp. 10024–10035.

[130] S. Riniker and G. A. Landrum. "Better informed distance geometry: using what we know to improve conformation generation". In: *Journal of chemical information and modeling* 55.12 (2015), pp. 2562–2574.

[131] S. Wills, R. Sanchez-Garcia, T. Dudgeon, S. D. Roughley, A. Merritt, R. E. Hubbard, J. Davidson, F. von Delft, and C. M. Deane. "Fragment Merging Using a Graph Database Samples Different Catalogue Space than Similarity Search". In: *Journal of Chemical Information and Modeling* (2023).

[132] A. B. Deore, J. R. Dhumane, R. Wagh, and R. Sonawane. "The stages of drug discovery and development process". In: *Asian Journal of Pharmaceutical Research and Development* 7.6 (2019), pp. 62–67.

[133] L. Hu, M. L. Benson, R. D. Smith, M. G. Lerner, and H. A. Carlson. "Binding MOAD (mother of all databases)". In: *Proteins: Structure, Function, and Bioinformatics* 60.3 (2005), pp. 333–340.

[134] P. G. Francoeur, T. Masuda, J. Sunseri, A. Jia, R. B. Iovanisci, I. Snyder, and D. R. Koes. "Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design". In: *Journal of chemical information and modeling* 60.9 (2020), pp. 4200–4215.

[135] A. Schneuing, Y. Du, C. Harris, A. R. Jamasb, I. Igashov, T. L. Blundell, P. Lio, C. P. Gomes, M. Welling, M. M. Bronstein, et al. "Structure-based Drug Design with Equivariant Diffusion Models". In: (2022).

[136] A. Alhossary, S. D. Handoko, Y. Mu, and C.-K. Kwoh. "Fast, accurate, and reliable molecular docking with QuickVina 2". In: *Bioinformatics* 31.13 (2015), pp. 2214–2216.

[137] G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins. "Quantifying the chemical beauty of drugs". In: *Nature chemistry* 4.2 (2012), pp. 90–98.

[138] P. Ertl and A. Schuffenhauer. "Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions". In: *Journal of cheminformatics* 1 (2009), pp. 1–11.

[139] X. Peng, S. Luo, J. Guan, Q. Xie, J. Peng, and J. Ma. "Pocket2mol: Efficient molecular sampling based on 3d protein pockets". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 17644–17655.

[140] C. A. Lipinski. "Lead-and drug-like compounds: the rule-of-five revolution". In: *Drug discovery today: Technologies* 1.4 (2004), pp. 337–341.

[141] T. T. Tanimoto. *Elementary mathematical theory of classification and prediction*. International Business Machines Corp., 1958.

[142] D. Bajusz, A. Rácz, and K. Héberger. "Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations?" In: *Journal of cheminformatics* 7.1 (2015), pp. 1–13.

[143] J. Song, C. Meng, and S. Ermon. "Denoising diffusion implicit models". In: *arXiv preprint arXiv:2010.02502* (2020).

[144] Y.-L. Liao, B. M. Wood, A. Das, and T. Smidt. "EquiformerV2: Improved Equivariant Transformer for Scaling to Higher-Degree Representations". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=mCOBKZmrzD.

[145] C. Harris, K. Didi, A. R. Jamasb, C. K. Joshi, S. V. Mathis, P. Lio, and T. Blundell. "Benchmarking Generated Poses: How Rational is Structure-based Drug Design with Generative Models?" In: *arXiv preprint arXiv:2308.07413* (2023).

[146] A. Morehead and J. Cheng. "Geometry-Complete Perceptron Networks for 3D Molecular Graphs". In: *Bioinformatics* (2024). Also presented at the AAAI 2023 DLG and AI2ASE workshops.

[147] J. Ho, A. Jain, and P. Abbeel. "Denoising diffusion probabilistic models". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851.

[148] M. H. Segler, T. Kogej, C. Tyrchan, and M. P. Waller. "Generating focused molecule libraries for drug discovery with recurrent neural networks". In: *ACS central science* 4.1 (2018), pp. 120–131.

[149] W. Jin, R. Barzilay, and T. Jaakkola. "Junction Tree Variational Autoencoder for Molecular Graph Generation". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 2323–2332. URL: https://proceedings.mlr.press/v80/jin18a.html.

[150] A. Dhakal, C. McKay, J. J. Tanner, and J. Cheng. "Artificial intelligence in the prediction of protein–ligand interactions: recent advances and future directions". In: *Briefings in Bioinformatics* 23.1 (2022), bbab476.

[151] W. Lu, Q. Wu, J. Zhang, et al. "Tankbind: Trigonometry-aware neural networks for drug-protein binding structure prediction". In: *Advances in neural information processing systems* 35 (2022), pp. 7236–7249.

[152] J. Eberhardt, D. Santos-Martins, A. F. Tillack, and S. Forli. "AutoDock Vina 1.2. 0: New docking methods, expanded force field, and python bindings". In: *Journal of chemical information and modeling* 61.8 (2021), pp. 3891–3898.

[153] C. Discovery, J. Boitreaud, J. Dent, et al. "Chai-1: Decoding the molecular interactions of life". In: *bioRxiv* (2024), pp. 2024–10.

[154] J. Wohlwend, G. Corso, S. Passaro, et al. "Boltz-1: Democratizing Biomolecular Interaction Modeling". In: *bioRxiv* (2024), pp. 2024–11.

[155] R. T. Chen and Y. Lipman. "Flow matching on general geometries". In: *The Twelfth International Conference on Learning Representations*. 2024.

[156] A. Tong, K. Fatras, N. Malkin, and others. "Improving and generalizing flow-based generative models with minibatch optimal transport". In: *Transactions on Machine Learning Research* (2024). Expert Certification. ISSN: 2835-8856. URL: https://openreview.net/forum?id=CD9Snc73AW.

[157] T. Karras, M. Aittala, T. Aila, and S. Laine. "Elucidating the design space of diffusion-based generative models". In: *Advances in neural information processing systems* 35 (2022), pp. 26565–26577.

[158] P. Esser, S. Kulal, A. Blattmann, et al. "Scaling rectified flow transformers for high-resolution image synthesis". In: *Forty-first International Conference on Machine Learning*. 2024.

[159] L. Klein, A. Krämer, and F. Noé. "Equivariant flow matching". In: *Advances in Neural Information Processing Systems* 36 (2024).

[160] CASP16-Organizers. "CASP16 Abstracts". In: *CASP16* (2024). URL: https://predictioncenter.org/casp16/doc/CASP16_Abstracts.pdf#page=171.08.

[161] G. Corso, V. R. Somnath, N. Getz, et al. "Flexible Docking via Unbalanced Flow Matching". In: *ICML Workshop ML for Life and Material Science: From Theory to Industry Applications*. 2024.

[162] B. Jing, B. Berger, and T. Jaakkola. "AlphaFold Meets Flow Matching for Generating Protein Ensembles". In: *Forty-first International Conference on Machine Learning*. 2024.

[163] E. Mathieu and M. Nickel. "Riemannian continuous normalizing flows". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 2503–2515.

[164] J. Bose, T. Akhound-Sadegh, G. Huguet, et al. "SE(3)-Stochastic Flow Matching for Protein Backbone Generation". In: *The Twelfth International Conference on Learning Representations*. 2024.

[165] G. Papamakarios, E. Nalisnick, D. J. Rezende, et al. "Normalizing flows for probabilistic modeling and inference". In: *Journal of Machine Learning Research* 22.57 (2021), pp. 1–64.

[166] J. Dauparas, I. Anishchenko, N. Bennett, et al. "Robust deep learning–based protein sequence design using ProteinMPNN". In: *Science* 378.6615 (2022), pp. 49–56.

[167] H. Stark, B. Jing, R. Barzilay, and T. Jaakkola. "Harmonic Self-Conditioned Flow Matching for joint Multi-Ligand Docking and Binding Site Design". In: *Forty-first International Conference on Machine Learning*. 2024.

[168] G. Corso. *Modeling molecular structures with intrinsic diffusion models*. Massachusetts Institute of Technology, 2023.

[169] A.-A. Pooladian, H. Ben-Hamu, C. Domingo-Enrich, et al. "Multisample Flow Matching: Straightening Flows with Minibatch Couplings". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 28100–28127.

[170] G. Corso, A. Deng, N. Polizzi, R. Barzilay, and T. S. Jaakkola. "Deep Confident Steps to New Pockets: Strategies for Docking Generalization". In: *The Twelfth International Conference on Learning Representations*. 2024.

[171] Y. Zhang and J. Skolnick. "Scoring function for automated assessment of protein structure template quality". In: *Proteins: Structure, Function, and Bioinformatics* 57.4 (2004), pp. 702–710.

[172] A. Morehead, N. Giri, J. Liu, and J. Cheng. "Deep Learning for Protein-Ligand Docking: Are We There Yet?" In: *ICML AI4Science Workshop*. Selected as a spotlight presentation. 2024.

[173] P. D. Bank. "Protein data bank". In: *Nature New Biol* 233.223 (1971), pp. 10–1038.

[174] Z. Liu, Y. Li, L. Han, et al. "PDB-wide collection of binding data: current status of the PDBbind database". In: *Bioinformatics* 31.3 (2015), pp. 405–412.

[175] P. Eastman, J. Swails, J. D. Chodera, et al. "OpenMM 7: Rapid development of high performance algorithms for molecular dynamics". In: *PLoS computational biology* 13.7 (2017), e1005659.

[176] J. Durairaj, Y. Adeshina, Z. Cao, et al. "PLINDER: The protein-ligand interactions dataset and evaluation resource". In: *ICML Workshop ML for Life and Material Science: From Theory to Industry Applications*. 2024.

[177] M. J. Hartshorn, M. L. Verdonk, G. Chessari, et al. "Diverse, high-quality test set for the validation of protein- ligand docking performance". In: *Journal of medicinal chemistry* 50.4 (2007), pp. 726–741.

[178] G. L. Warren, T. D. Do, B. P. Kelley, A. Nicholls, and S. D. Warren. "Essential considerations for using protein–ligand structures in drug discovery". In: *Drug Discovery Today* 17.23-24 (2012), pp. 1270–1281.

[179] X. Du, Y. Li, Y.-L. Xia, S.-M. Ai, J. Liang, P. Sang, X.-L. Ji, and S.-Q. Liu. "Insights into protein–ligand interactions: mechanisms, models, and methods". In: *International journal of molecular sciences* 17.2 (2016), p. 144.

[180] L. A. Abriata. "The Nobel Prize in Chemistry: past, present, and future of AI in biology". In: *Communications Biology* 7.1 (2024), p. 1409.

[181] G. Corso, A. Deng, B. Fry, N. Polizzi, R. Barzilay, and T. Jaakkola. *The Discovery of Binding Modes Requires Rethinking Docking Generalization*. Feb. 2024. URL: https://doi.org/10.5281/zenodo.10656052.

[182] J. Yim, H. Stärk, G. Corso, B. Jing, R. Barzilay, and T. S. Jaakkola. "Diffusion models in protein structure and docking". In: *Wiley Interdisciplinary Reviews: Computational Molecular Science* 14.2 (2024), e1711.

[183] X. Zhang, O. Zhang, C. Shen, W. Qu, S. Chen, H. Cao, Y. Kang, Z. Wang, E. Wang, J. Zhang, et al. "Efficient and accurate large library ligand docking with KarmaDock". In: *Nature Computational Science* 3.9 (2023), pp. 789–804.

[184] M. Masters, A. Mahmoud, and M. Lill. "Fusiondock: Physics-informed diffusion model for molecular docking". In: *ICML2023 CompBio Workshop*. 2023.

[185] M. Plainer, M. Toth, S. Dobers, H. Stark, G. Corso, C. Marquet, and R. Barzilay. "DiffDock-Pocket: Diffusion for Pocket-Level Docking with Sidechain Flexibility". In: *NeurIPS 2023 Machine Learning in Structural Biology Workshop* (2023).

[186] H. Guo, S. Liu, H. Mingdi, Y. Lou, and B. Jing. "DiffDock-Site: A Novel Paradigm for Enhanced Protein-Ligand Predictions through Binding Site Identification". In: *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*. 2023.

[187] Q. Pei, K. Gao, L. Wu, J. Zhu, Y. Xia, S. Xie, T. Qin, K. He, T.-Y. Liu, and R. Yan. "FABind: Fast and accurate protein-ligand binding". In: *Advances in Neural Information Processing Systems* 36 (2024).

[188] J. Zhu, Z. Gu, J. Pei, and L. Lai. "DiffBindFR: An SE (3) Equivariant Network for Flexible Protein-Ligand Docking". In: *Chemical Science* (2024).

[189] D. Cao, M. Chen, R. Zhang, Z. Wang, M. Huang, J. Yu, X. Jiang, Z. Fan, W. Zhang, H. Zhou, et al. "SurfDock is a surface-informed diffusion generative model for reliable and accurate protein–ligand complex prediction". In: *Nature Methods* (2024), pp. 1–13.

[190] Y. Huang, O. Zhang, L. Wu, C. Tan, H. Lin, Z. Gao, S. Li, S. Li, et al. "Re-Dock: Towards Flexible and Realistic Molecular Docking with Diffusion Bridge". In: *arXiv preprint arXiv:2402.11459* (2024).

[191] R. Miñán, J. G. Sáenz, A. Molina, et al. "GeoDirDock: Guiding Docking Along Geodesic Paths". In: *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*.

[192] A. Morehead and J. Cheng. "FlowDock: Geometric Flow Matching for Generative Protein-Ligand Docking and Affinity Prediction". In: *Intelligent Systems for Molecular Biology (ISMB)*. 2025.

[193] G. Corso, V. R. Somnath, N. Getz, R. Barzilay, T. Jaakkola, and A. Krause. "Flexible docking via unbalanced flow matching". In: *ICML'24 Workshop ML for Life and Material Science: From Theory to Industry Applications*. 2024.

[194] Z. Qiao, F. Ding, T. Dresselhaus, M. A. Rosenfeld, X. Han, O. Howell, A. Iyengar, S. Opalenski, A. S. Christensen, S. K. Sirumalla, et al. "NeuralPLexer3: Physio-Realistic Biomolecular Complex Structure Prediction with Flow Models". In: *arXiv preprint arXiv:2412.10743* (2024).

[195] Y. Yu, S. Lu, Z. Gao, H. Zheng, and G. Ke. "Do deep learning models really outperform traditional approaches in molecular docking?" In: *ICLR 2023-Machine Learning for Drug Discovery workshop*.

[196] D. Errington, C. Schneider, C. Bouysset, and F. A. Dreyer. "Assessing interaction recovery of predicted protein-ligand poses". In: *NeurIPS Workshop Foundation Models for Science: Progress, Opportunities, and Challenges*. 2024.

[197] A. N. Jain, A. E. Cleves, and W. P. Walters. "Deep-Learning Based Docking Methods: Fair Comparisons to Conventional Docking Workflows". In: *arXiv preprint arXiv:2412.02889* (2024).

[198] D. A. Sharon, Y. Huang, M. Oyewole, and S. Mustafa. "How to Go With the Flow: an Analysis of Flow Matching Molecular Docking Performance With Priors of Varying Information Content". In: *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*. 2024.

[199] Q. Hu, Z. Wang, J. Meng, W. Li, J. Guo, Y. Mu, S. Wang, L. Zheng, and Y. Wei. "OpenDock: a pytorch-based open-source framework for protein–ligand docking and modelling". In: *Bioinformatics* 40.11 (2024), btae628.

[200] X. Robin, G. Studer, J. Durairaj, J. Eberhardt, T. Schwede, and W. P. Walters. "Assessment of protein–ligand complexes in CASP15". In: *Proteins: Structure, Function, and Bioinformatics* 91.12 (2023), pp. 1811–1821.

[201] P. Eastman and V. Pande. "OpenMM: A hardware-independent framework for molecular simulations". In: *Computing in science & engineering* 12.4 (2010), pp. 34–39.

[202] H. Cheng, R. D. Schaeffer, Y. Liao, L. N. Kinch, J. Pei, S. Shi, B.-H. Kim, and N. V. Grishin. "ECOD: an evolutionary classification of protein domains". In: *PLoS computational biology* 10.12 (2014), e1003926.

[203] M. Steinegger and J. Söding. "MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets". In: *Nature biotechnology* 35.11 (2017), pp. 1026–1028.

[204] Z. Liu, M. Su, L. Han, J. Liu, Q. Yang, Y. Li, and R. Wang. "Forging the basis for developing protein–ligand interaction scoring functions". In: *Accounts of chemical research* 50.2 (2017), pp. 302–309.

[205] W. L. DeLano et al. "Pymol: An open-source molecular graphics tool". In: *CCP4 Newsl. Protein Crystallogr* 40.1 (2002), pp. 82–92.

[206] C. Bouysset and S. Fiorucci. "ProLIF: a library to encode molecular interactions as fingerprints". In: *Journal of cheminformatics* 13.1 (2021), p. 72.

[207] V. Mariani, M. Biasini, A. Barbato, and T. Schwede. "lDDT: a local superposition-free score for comparing protein structures and models using distance difference tests". In: *Bioinformatics* 29.21 (2013), pp. 2722–2728.

[208] Y. Rubner, C. Tomasi, and L. J. Guibas. "The earth mover's distance as a metric for image retrieval". In: *International journal of computer vision* 40 (2000), pp. 99–121.

[209] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". In: *Nature methods* 17.3 (2020), pp. 261–272.

[210] O. Trott and A. J. Olson. "AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading". In: *Journal of computational chemistry* 31.2 (2010), pp. 455–461.

[211] R. Krivák and D. Hoksza. "P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure". In: *Journal of cheminformatics* 10 (2018), pp. 1–12.

[212] Z. Guo, J. Liu, Y. Wang, M. Chen, D. Wang, D. Xu, and J. Cheng. "Diffusion models in bioinformatics and computational biology". In: *Nature reviews bioengineering* 2.2 (2024), pp. 136–154.

[213] G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. S. Jaakkola. "DiffDock: Diffusion Steps, Twists, and Turns for Molecular Docking". In: *The Eleventh International Conference on Learning Representations*. 2023.

[214] R. S. Roy, J. Liu, N. Giri, Z. Guo, and J. Cheng. "Combining pairwise structural similarity and deep learning interface contact prediction to estimate protein complex model accuracy in CASP15". In: *Proteins: Structure, Function, and Bioinformatics* 91.12 (2023), pp. 1889–1902.

[215] D. Kingma, T. Salimans, B. Poole, and J. Ho. "Variational diffusion models". In: *Advances in neural information processing systems* 34 (2021), pp. 21696–21707.

[216] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. "Flow Matching for Generative Modeling". In: *The Eleventh International Conference on Learning Representations*. 2023.

[217] C. McInnes. "Virtual screening strategies in drug discovery". In: *Current opinion in chemical biology* 11.5 (2007), pp. 494–502.

[218] T. S. Cohen and M. Welling. "Steerable CNNs". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=rJQKYt5ll.

[219] F. Fuchs, D. Worrall, V. Fischer, and M. Welling. "SE(3)-Transformers: 3D Roto-Translation Equivariant Attention Networks". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 1970–1981. URL: https://proceedings.neurips.cc/paper/2020/file/15231a7ce4ba789d13b7Paper.pdf.

[220] A. Gu, C. Gulcehre, T. Paine, M. Hoffman, and R. Pascanu. "Improving the gating mechanism of recurrent neural networks". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 3800–3809.

[221] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[222] Y. Liu, L. Wang, M. Liu, Y. Lin, X. Zhang, B. Oztekin, and S. Ji. "Spherical Message Passing for 3D Molecular Graphs". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=givsRXsOt9r.

[223] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 801–818.

[224] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[225] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[226] W. Falcon et al. "PyTorch Lightning". In: *GitHub* 3 (2019). URL: https://github.com/PyTorchLightning/pytorch-lightning.

[227] J. Brandstetter, R. Hesselink, E. van der Pol, E. J. Bekkers, and M. Welling. "Geometric and Physical Quantities improve E(3) Equivariant Message Passing". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=_xwr8gOBeV1.

[228] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. "Deep sets". In: *Advances in neural information processing systems* 30 (2017).

[229] P. Hermosilla, M. Schäfer, M. Lang, G. Fackelmann, P.-P. Vázquez, B. Kozlikova, M. Krone, T. Ritschel, and T. Ropinski. "Intrinsic-Extrinsic Convolution and Pooling for Learning on 3D Protein Structures". In: *International Conference on Learning Representations*. 2021.

[230] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International Conference on Machine Learning*. PMLR. 2015, pp. 2256–2265.

[231] J. Köhler, L. Klein, and F. Noé. "Equivariant flows: exact likelihood generative learning for symmetric densities". In: *International conference on machine learning*. PMLR. 2020, pp. 5361–5370.

[232] W. P. Walters and M. Murcko. "Assessing the impact of generative AI on medicinal chemistry". In: *Nature biotechnology* 38.2 (2020), pp. 143–145.

[233] F. Urbina, F. Lentzos, C. Invernizzi, and S. Ekins. "Dual use of artificial-intelligence-powered drug discovery". In: *Nature Machine Intelligence* 4.3 (2022), pp. 189–191.

[234] S. Elfwing, E. Uchibe, and K. Doya. "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning". In: *Neural Networks* 107 (2018), pp. 3–11.

[235] I. Loshchilov and F. Hutter. "Decoupled weight decay regularization". In: *arXiv preprint arXiv:1711.05101* (2017).

[236] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems* 32 (2019).

[237] M. Fey and J. E. Lenssen. "Fast graph representation learning with PyTorch Geometric". In: *arXiv preprint arXiv:1903.02428* (2019).

[238] O. Yadan. *Hydra - A framework for elegantly configuring complex applications*. Github. 2019. URL: `https://github.com/facebookresearch/hydra`.

[239] A. Morehead, J. Liu, P. Neupane, N. Giri, and J. Cheng. "Protein-ligand structure and affinity prediction in CASP16 using a geometric deep learning ensemble and flow matching". In: *Proteins: Structure, Function, and Bioinformatics* (2025).

[240] A. Morehead, N. Giri, J. Liu, P. Neupane, and J. Cheng. *Deep Learning for Protein-Ligand Docking: Are We There Yet?* Version 1.2.0. Feb. 2025. URL: https://doi.org/10.5281/zenodo.14629652.

[241] M. Akhtar, O. Benjelloun, C. Conforti, J. Giner-Miguelez, N. Jain, M. Kuchnik, Q. Lhoest, P. Marcenac, M. Maskey, P. Mattson, L. Oala, P. Ruyssen, R. Shinde, E. Simperl, G. Thomas, S. Tykhonov, J. Vanschoren, S. Vogler, and C.-J. Wu. *Croissant: A Metadata Format for ML-Ready Datasets*. 2024. arXiv: 2403.19546 [cs.LG].

[242] M. Buttenschoen, G. M. Morris, and C. M. Deane. *PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences*. Aug. 2023. URL: https://doi.org/10.48550/arXiv.2308.05777.

# VITA

Alex Morehead earned his B.S. in Computer Science with highest distinction from Missouri Western State University in 2020. He then pursued a Ph.D. in Computer Science at the University of Missouri-Columbia, earning his M.S. in 2024 and Ph.D. in 2025. His research focused on deep learning for representation learning and generative modeling of proteins and other 3D biomolecules, leading to 20 peer-reviewed publications in prestigious venues such as ICLR, ISMB, Nature Communications Chemistry, Bioinformatics, and Proteins.

His work gained recognition in competitive settings, including the 16$^{\text{th}}$ Critical Assessment of Techniques for Structure Prediction (CASP16), where his method MULTICOM_LIGAND ranked 5$^{\text{th}}$ in ligand structure and binding affinity prediction, earning an oral presentation. Additionally, his method GCPNet-EMA served as a core component of MULTICOM_GATE, which placed 2$^{\text{nd}}$ in protein complex structure quality assessment. Further, his Ph.D. research was awarded Berkeley Lab's 2025 Admiral Grace Hopper Postdoctoral Fellowship in Computing Sciences.

Alex's research interests include developing AI-driven algorithms for the physical and life sciences and applying these algorithms to downstream tasks such as biomolecular design in therapeutics, energy research, and climate science. He specializes in geometric deep learning and generative modeling for protein-protein and protein-ligand interactions, inverse folding, and drug design. His interdisciplinary background further spans synthetic biology, edge computing, and urban data science, reinforcing his commitment to advancing AI for scientific discovery.