Machine Learning Methods for Protein Model Quality Estimation

Md Hossain Shuvo

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Applications

Debswapna Bhattacharya, Chair
Lenwood S. Heath
Alexey V. Onufriev
Liqing Zhang
Scott Emrich

December 8, 2023
Blacksburg, Virginia

Keywords: Model quality estimation, Protein-protein interaction, Deep learning, Graph representation learning, Protein structure prediction

Machine Learning Methods for Protein Model Quality Estimation

Md Hossain Shuvo

(ABSTRACT)

Protein model quality estimation helps at various stages of the computational prediction of protein three-dimensional (3D) structures, which are essential for numerous biological processes. Existing quality estimation methods for monomeric and multimeric protein models fall short of leveraging the latest technological breakthrough in deep learning that has recently revolutionized protein structure prediction, particularly in harnessing the power of a symmetry-aware network informed by the Protein Language Model (pLM) and the integration of important evolutionary and structural information, limiting their ability to improve quality estimation performance. Moreover, despite the significant progress made by AlphaFold2 in predicting highly accurate protein models, existing methods fall short of the estimation of high-resolution protein model quality and its application in improving the accuracy of predicted protein models. Therefore, the objectives of my research include 1) the development of quality estimation methods for monomeric protein models by learning efficient structural representation with the application of the latest deep learning architectures; 2) the application of our quality estimation methods to improve the structural accuracy of computationally generated protein models; and 3) the development of multimeric protein model quality estimation methods with deep graph learning of protein-protein interface. The benefits of achieving these objectives are two-fold for the broader scientific community. Firstly, it will enable the evaluation of protein models without the presence of experimentally determined protein structures. Secondly, our protein model quality estimation methods can potentially guide the overall structure prediction process and improve the structural accuracy of predicted models.

Machine Learning Methods for Protein Model Quality Estimation

Md Hossain Shuvo

(GENERAL AUDIENCE ABSTRACT)

In my research, I developed protein model quality estimation methods aimed at evaluating the reliability of computationally predicted protein models in the absence of experimentally solved ground truth structures. These methods specifically focus on estimating errors within the protein models to quantify their structural accuracy. Recognizing that even the most advanced protein structure prediction techniques may produce models with errors, I also developed a complementary protein model refinement method. This refinement method iteratively optimizes the weakly modeled regions, guided by the error estimation module of my quality estimation approach. The development of these model quality estimation methods, therefore, not only offers valuable insights into the structural reliability of protein models but also contributes to optimizing the overall reliability of protein models generated by state-of-the-art computational methods.

Dedication

The dissertation is dedicated to my mother and in loving memory of my father

Acknowledgments

I would like to thank my advisor Dr. Debswapna Bhattacharya for the guidance and support from the beginning of my Ph.D. studies. You have set an example of excellence as a researcher, mentor, instructor, and role model. I would also like to Dr. Lenwood S. Heath, Dr. Alexey V. Onufriev, Dr. Liqing Zhang, and Dr. Scott Emrich for serving on my Dissertation Committee and for their valuable insights and inputs to my research. Finally, I am thankful to my current and former colleagues in our lab for their collaboration.

Contents

List of Figures

List of Tables

List of Abbreviations

ρ Spearman Correlation Coefficient

τ Kendall's Tau Correlation Coefficient

r Pearson Correlation Coefficient

3D 3-Dimensional

AUC Area Under the Curve

CASP Critical Assessment of Protein Structure Prediction

CNN Convolutional Neural Network

EGNN Equivariant Graph Neural Network

ESM Evolutionary Scale Modeling

GAT Graph Attention Network

GCN Graph Convolutional Network

GDT Global Distance Test

GNN Graph Neural Network

GTN Graph Transformer Network

HHblits HMM-HMM-based lightning-fast iterative sequence search

HMM Hidden Markov Model

LSTM Long Short-Term Memory

MQA Model Quality Assessment

MSA Multiple Sequence Alignment

NEFF Number of Effective Sequences

NR Non-Redundant

PDB Protein Data Bank

pLM Protein Language Model

ResNet Residual Neural Network

RF Random Forest

RMSD Root Mean Square Deviation

ROC Receiver Operating Characteristics

SA Solvent Accessibility

SS Secondary Structure

SVM Support Vector Machine

UniRef UniProt Reference Cluster

Chapter 1

Introduction

1.1 Protein structure prediction

Proteins are the essential macromolecules that catalyze various biological processes including gene expression and regulation, signal transduction, and chemical reactions in living systems [1, 2]. Existing protein structure determination approaches include X-ray crystallography, electron microscopy (EM), and nuclear magnetic resonance (NMR) [3, 4]. However, experimental approaches are time-consuming, expensive, and require extensive human effort, leading to a substantial gap between the number of known protein sequences and experimentally solved proteins [5, 6]. Therefore, computational approaches to protein structure prediction offer an alternative strategy to experimental protein structure prediction [7]. Recent technological breakthroughs have fueled the development of protein structure prediction methods with AlphaFold2, RoseTTaFold, ESMFold, HelixFold-Single, and OmegaFold demonstrating complementary strengths and delivering highly accurate protein structures.[2, 8, 9, 10, 11].

1.2 Protein model quality estimation

Protein model quality estimation is the problem of evaluating the quality of the computationally predicted structural models in the absence of experimentally determined structures [12]. Methods for quality estimation of computationally predicted protein structural models serve as a key component of protein structure prediction [13, 14] in validating and evaluating

predicted protein models at multiple stages of a structure prediction pipeline thus greatly affecting its prediction accuracy [15, 16]. The computational prediction of protein 3D structures is cost-effective and faster compared to the experimental strategies [8, 17] and there has been significant progress in developing computational approaches, achieving state-of-the-art accuracy [2, 9, 18]. Despite the progress and recent breakthrough by AlphaFold2 [2] in protein structure prediction, the quality of the predicted structure can be very variable, thus prompting uncertainty about their reliability [19]. Quality estimation for predicted protein models (a.k.a. protein model quality estimation) can accurately inform about their reliable and unreliable regions through the accurate estimation of structural error from the experimental structure [14, 20, 21] essentially informing about the reliability of the predicted structure. The estimation of structural error helps in further improving their structural quality [20, 22, 23] essentially improving the structural reliability of predicted protein models through iterative structural refinement.

1.3 Approaches to protein model quality estimation

Approaches to protein model quality estimation methods are broadly categorized into 'single-model' methods and 'consensus' approaches. The single-model method estimates the quality of protein models based on the model itself [14, 20, 21]. On, the other hand, consensus approaches (a.k.a. multi-model approach) depend on a pool of models to score the quality of each of the models in the pool [24, 25, 26]. Therefore, consensus methods are not widely applicable, specifically when there is a lack of a large pool of models. Single-model methods, however, are free from such a limitation and can be independently employed for scoring and model selection.

Single-model methods can be broadly categorized into two major classes: monomeric protein model quality estimation and multimeric protein complex model quality estimation.

Monomeric protein quality estimation methods [14, 20, 21], assess the quality of protein models with a single chain. On the other hand, multimeric model quality estimation methods [27, 28, 29], evaluate the quality of protein complexes with multiple subunits. Techniques for both monomeric and multimer protein complex quality estimation, play integral parts in the prediction of protein monomeric and protein multimeric complex prediction pipelines, respectively.

1.4 Deep learning in protein model quality estimation

Methods for both monomeric and multimeric model quality estimation are increasingly adopting the recent advancements in deep learning architectures, delivering improved quality estimation performance. For instance, Ornate[30], 3DCNN [31], ProQ4 [32], DOVE [33], leverage deep Convolutional Neural Network (CNN) [34]. Several protein model quality estimation methods including QDeep [21], DeepACCNet [20], ResNetQA [12], DeepUMQA [35] leverage ResNet [36] which has emerged as the improvement over CNN and transformed the training of deep Convolutional Neural Networks. Recent advancement in several Graph Neural Networks (GNN)-based architecture has also gained significant attention in the domain of protein model quality estimation problems as proteins can inherently be represented by graphs. Both monomeric and multimeric protein model quality estimation methods including GraphQA [37], ProteinGCN [38], PIQLE [27], VoroIF_GNN [29], GNN-DOVE [39], TRscore [28], GDockScore [40], DeepRank-GNN-esm [41], EnQA [42] leverage GNN-based architecture including Graph Convolutional Network (GCN) [43], Graph Attention Network, (GAT)[44] Graph Transformer Network (GTN), [45] and Equivariant Graph Neural Network (EGNN) [46]. Overall the application of the latest deep learning architectures improves the quality estimation performance as demonstrated by these existing approaches to protein monomeric and multimeric quality estimation.

1.5 Limitation and challenges to protein model quality estimation

State-of-the-art protein model quality estimation methods adopt various strategies to evaluate the quality of both the monomeric and multimeric structures. For instance, methods for monomeric protein models use various combinations of features and employ different machine-learning approaches including Support Vector Machine (SVM) and Random Forest for estimating the quality of a protein model [47, 48, 49]. In addition to these traditional machine-learning-based methods, a growing number of approaches are employing deep learning including multi-layer perceptron and convolutional neural networks (CNNs) [30, 31]

Additionally, methods for multimeric protein complexes estimate the quality by evaluating the accuracy of the interface region of the interacting proteins within the complex. While estimating the quality, existing approaches learn the voxelized representation of the protein-protein interface with the application of convolutional neural networks (CNNs) [28, 50]. Recently, representation learning with graph neural networks [51] is gaining significant attention, leading to the development of several protein complex model quality estimation methods by applying graph attention network (GAT) [39, 44] and graph transformer networks (GTN) [39, 45].

Despite their effectiveness, these existing approaches have several limitations. For instance, single-model quality estimation methods for monomeric protein models do not consider some key factors including the incorporation of inter-residue distance [52] information and an ensemble learning of very deep residual neural networks (ResNets) [36] that has emerged as a breakthrough deep learning technology

Additionally, state-of-the-art quality estimation methods for multimeric protein complexes have two major shortcomings. First, they do not consider several key information for efficient representation of multimeric protein models. For instance, they do not consider the geometry of the interaction interface that carries important structural information for protein complexes [53]. Additionally, most of the protein complex model scoring approaches typically rely on the physicochemical properties or energetic contributions of the interacting atoms but do not consider evolutionarily information in the form of multiple sequence alignments (MSAs) as well as the sequence. Second, while employing graph neural network-based models including Graph Convolutional Neural Network (GCN), Graph Attention Network (GAT), and Graph Transformer Network (GTN) that operate efficiently on protein graphs lacking a fixed order and providing important invariant properties regardless of node permutation, they do not exploit the symmetry-aware Equivariant Graph Neural Network (EGNN) that can inherently capture symmetry under certain rotations and translations when operating on 3D objects such as proteins, providing additional benefits including equivariance to rotation and translation while also maintaining invariance to node permutation.

1.6 Research objectives and outcomes

The primary objective of my research is to develop protein model quality estimation methods to accurately evaluate the structural quality of computationally generated protein models by addressing the limitations of existing protein model quality estimation methods with the application of data-driven machine learning approaches. Here we introduce novel techniques for estimating the quality of protein models that leverage the latest advancements in deep representation learning. Our approaches involve new strategies to model the structural characteristics of proteins, leading to better representation learning and resulting in improved performance in estimating the quality of both the monomeric and multimeric protein models.

In terms of the monomeric protein model quality estimation method, we present the distance-based method as the first approach to apply the power of an ensemble of very deep residual neural network architectures for performing residue-level error estimates to accurately predict the quality of their quality. We additionally, present a protein model refinement method by integrating the error estimation module of our monomeric protein model to quantify their unreliable regions and then applying an optimization algorithm to iteratively improve their structural quality.

In terms of the multimeric protein model quality estimation, we apply a deep graph neural network to develop a method for estimating the quality of multimeric protein complexes through the prediction of protein-protein interface quality. Ours is the first method to leverage multimeric interaction geometries and evolutionarily information along with sequence- and structure-derived features to estimate the protein-protein interface quality. We further develop an improved multimeric protein model quality estimation method by applying a symmetry-aware Equivariant Neural Network (EGNN) integrated with large Protein Language Model (pLM)-based embeddings.

We perform comprehensive benchmarking of our methods on benchmark datasets against the state-of-the-art competing methods. We additionally validate the efficacy of our methods through comprehensive testing in blind settings under the model quality assessment category of Critical Assessment of Protein Structure Prediction (CASP) experiments.

1.7 Dissertation outline and contributions

The document is structured as follows, covering five research outcomes from Chapter 2 to Chapter 6 and the Conclusion and Future plan in Chapter 7:

In **Chapter 2**, we present our **Research Outcome 1** of developing a quality estimation method for monomeric protein models by training an ensemble of very deep residual neural

networks (ResNets) with the incorporation of distance information, titled

"QDeep: distance-based protein model quality estimation by residue-level ensemble error classifications using stacked deep residual neural networks" [21] [**published in Bioinformatics**]

In **Chapter 3**, we present our **Research Outcome 2** of developing a high-resolution quality estimation method with an integrated scoring framework for evaluating monomeric protein structures, titled

iQDeep: an integrated web server for protein scoring using multiscale deep learning models" [54] [**Published in Journal of Molecular Biology**]

In **Chapter 4**, we present our **Resesarch Outcome 3** of developing a high-accuracy protein structure refinement method with the application of deep learning-based ensemble error estimates to improve the structural accuracy of monomeric protein models titled

"DeepRefiner: high-accuracy protein structure refinement by deep network calibration" [23] [**Published in Nucleic Acids Research**]

In **Chapter 5**, we present our **Resesarch Outcome 4** of developing a protein-protein interface quality estimation method for evaluating multimeric protein models by leveraging graph representation learning with the integration of interfacial geometries and evolutionarily information, titled

PIQLE: protein-protein interface quality estimation by deep graph learning of multimeric interaction geometries" [27] [**Published in Bioinformatics Advances**]

In **Chapter 6**, we present our **Resesarch Outcome 5** of developing an improved protein-protein interface quality estimation method for evaluating multimeric protein models by leveraging the symmetry-aware Equivariant Graph Neural Network (EGNN) integrated with protein language model-based ESM-2 embeddings, titled

"EquiRank: improved protein-protein interface quality estimation using protein-language-model-informed equivariant graph neural network"

In **Chapter 7**, we present a list of our publicly available scientific software disseminated to the broader scientific community, along with their usage and statistics.

In **Chapter 8**, we conclude by outlining our future plans.

Chapter 2

QDeep: distance-based protein model quality estimation by residue-level ensemble error classifications using stacked deep residual neural networks

2.1 Abstract

Protein model quality estimation, in many ways, informs protein structure prediction. Despite their tight coupling, existing model quality estimation methods do not leverage inter-residue distance information or the latest technological breakthrough in deep learning that has recently revolutionized protein structure prediction. We present a new distance-based single-model quality estimation method called QDeep by harnessing the power of stacked deep residual neural networks (ResNets). Our method first employs stacked deep ResNets to perform residue-level ensemble error classifications at multiple predefined error thresholds, and then combines the predictions from the individual error classifiers for estimating the quality of a protein structural model. Experimental results show that our method consistently outperforms existing state-of-the-art methods including ProQ2, ProQ3, ProQ3D, ProQ4, 3DCNN, MESHI, and VoroMQA in multiple independent test datasets across a wide-range of accuracy measures; and that predicted distance information significantly con-

tributes to the improved performance of QDeep. Availability and implementation: https://github.com/Bhattacharya-Lab/QDeep.

2.2 Introduction

Estimating the quality of a computationally generated protein structural model serves as a key component of protein structure prediction [13, 55]. Model quality estimation assists in validating and evaluating predicted protein models at multiple stages of a structure prediction pipeline, thus greatly affecting its prediction accuracy [15, 16] Methods for model quality estimation can be broadly categorized into two major classes that include 'single-model' methods and 'consensus' approaches. Single-model methods estimate structural quality purely from the model itself [14, 30, 31, 49, 55, 56, 57, 58] whereas consensus approaches exploit information from other models in a pool of possible alternatives [24, 25, 59, 60]. As such, performance of consensus approaches can be tremendously affected by the size and diversity of the model pool [13, 15, 61], sacrificing their generality and large-scale use in standalone structure prediction systems. Single-model methods, on the other hand, are free from such limitation and can be independently employed for scoring and model selection. As a result, singlemodel quality estimation methods are gaining increasing attention by the community in the recent editions of Critical Assessment of techniques for protein Structure Prediction (CASP) experiments [13, 59, 62], the universal standard for objectively evaluating the state-of-the-art of protein structure prediction.

Single-model quality estimation methods use various combinations of features and employ different machine-learning approaches for estimating the quality of a protein model without any knowledge of the experimental structure by learning a mapping from the features to its quality. For instance, ModelEvaluator [47] uses structural features to train support vector machine (SVM), RFMQA [48] utilizes structural features and potential energy terms for training random forest, ProQ2 [49] uses evolutionary sequence profile combined with contacts

and other structural features to train SVM. In addition to these traditional machine-learning-based methods, a growing number of approaches are employing deep learning, some of which delivering top performance in the most recent CASP13 experiment [13, 59]. For example, ProQ3D [14] and ProQ4 [32] exploit the strengths of multi-layer perceptron and 1D fully convolutional neural network (CNN), respectively. Other recent methods, such as Ornate [30] and 3DCNN [31, 56] take advantage of 3D CNNs, attaining state of-the-art performance. Despite their effectiveness, these approaches do not consider some key factors that can significantly improve single-model quality estimation performance. First, accurate prediction of inter-residue distance information has dramatically improved the nature of protein model generation [52, 63, 64] but none of the quality estimation methods incorporate distance information. Second, very deep and fully convolutional residual neural network (ResNet) [36] has emerged as a breakthrough deep learning technology that has revolutionized many computer vision tasks and very recently protein contact or distance prediction [65, 66], but their power has not yet been harnessed in estimating model quality. Third, most of these machine-leaning-based approaches typically rely on a single trained predictor for quality estimation either at the global or local level. That is, they do not make use of ensemble learning. In this article, we present a brand-new distance-based single-model quality estimation method QDeep by training an ensemble of stacked deep ResNets. Such architecture can perform residue-level ensemble error classifications at multiple predefined error thresholds. Here, we use 1, 2, 4, and 8Å as the error thresholds to model the GDT-TS score [67] by predicting the likelihood of the error between the Ca atom of any residue of a model to be within rÅ from the corresponding aligned residue in the experimental structure, where $r \epsilon \{1, 2, 4, 8\}$. Combined predictions from the individual error classifiers can then be used to estimate the quality of a protein structural model. We train QDeep using a redundancy-removed set of proteins from CASP9 and CASP10, validate the performance of the individual deep ResNet models on CASP11, and then evaluate its quality estimation performance on CASP12 and CASP13 targets. Our experimental results show that our method yields much better perfor-

mance than existing methods and also results in better discrimination between 'good' and 'bad' models. The improved performance of QDeep is deeply driven by our effective integration of distance information for single-model quality estimation. QDeep is freely available at https://github.com/Bhattacharya-Lab/QDeep.

2.3 Materials and Method

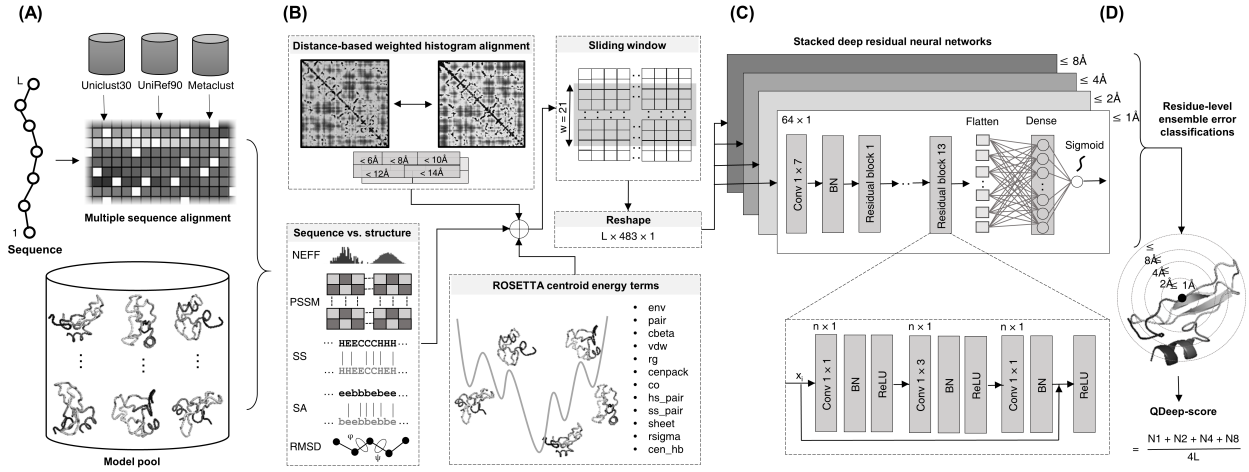

Figure 2.1: Flowchart of QDeep. (A) Multiple sequence alignment generation. (B) Distance-based, sequence versus structure consistency-based and ROSETTA centroid energy terms-based features collection. (C) Architecture of stacked deep ResNet classifiers at 1, 2, 4, and 8Å error thresholds. (D) Residue-level ensemble error classifications and their combination for model quality estimation.

The flowchart of QDeep is shown in Figure 2.1, which consists of four steps: multiple sequence alignment generation, feature collection, stacked deep ResNets training, and residue-level ensemble error classifications and their combination for model quality estimation.

2.3.1 Multiple sequence alignment generation

We generate multiple sequence alignment (MSA) (Fig. 2A) using HHblits [68] with a query sequence coverage of 10% and pairwise sequence identity of 90% against whole-genome sequence database Uniclust30 [69] for three iterations with an E-value inclusion threshold of

103. We also experiment the inclusion of other whole-genome sequence database UniRef90 [70] and metagenome database Metaclust [71] using the DeepMSA [72] pipeline to generate more sensitive and diverse MSA with improved coverage and alignment depth. The generated MSA serves as the key input to inter-residue distance prediction as well as other sequence-based features, such as secondary structure and solvent accessibility.

2.3.2 Feature collection

As shown in Figure 2B, we generate a total of 23 features for describing each residue of a model that includes distance-based weighted histogram alignment, sequence versus structure consistency, and ROSETTA centroid energy terms. We briefly describe them below.

Distance-based weighted histogram alignment

We predict inter-residue distance map of the target protein by feeding the MSA to DMPfold [52] and obtain the initial distance prediction without any iterative refinement (i.e., rawdistpred.current files) containing 20 distance bins with associated likelihoods between the interacting residue pairs. The distance map is then converted to 5 evenly distributed distance intervals: 6, 8, 10, 12, and 14Åby summing up likelihoods for distance bins below specific distance thresholds and considering only the residue pairs having likelihoods of at least 0.2 to reduce noise. We calculate observed inter-residue distance histogram for each model in the model pool at the same 5 distance intervals mentioned above to perform dynamic programming alignments of the predicted and observed distance histograms through eigen-decomposition [73] 2010). The 5 alignment scores, each of which ranges between 0 and 1, are used as 5 distance-based features after multiplying with empirically selected weights of 0.10, 0.25, 0.30, 0.25, and 0.10 for alignment scores at distance bins 6, 8, 10, 12, and 14Å, respectively, to allow higher emphasis on the distance intervals between 8 and 12Å.

Sequence versus structure consistency

Number of effective sequences: We use the normalized number of effective sequences (NEFF) [72] as a feature. NEFF is calculated as the reciprocated sum of the number of sequences in the MSA with a sequence identity >80% to the n^{th} sequence, divided by the total number of sequences in the MSA.

Sequence profile conservation score: We generate a sequence profile by searching the NR database using PSI-BLAST v2.2.26 software [74] with an E-value of 0.001. The information per position score from the resulting position-specific scoring matrix is then used as a feature after applying sigmoidal transformation to scale the score between 0 and 1.

Secondary structure (SS) and solvent accessibility (SA): We predict SS and SA using SPIDER3 [75] and use residue-specific binary agreement between predicted and observed SS as well as the squared error between predicted and observed relative SA as features.

Angular root mean square deviation (RMSD): We use normalized RMSD between the two backbone dihedral angles (ϕ, ψ) predicted from the sequence using SPIDER3 and their observed values computed from the models as two features, which are computed as:

$$AngularRMSD = \sqrt{\frac{1}{n}\sum_{i}(min(x_{2i} - x_{i1}, 2\pi - |x_2 i - x_i 1|))^2} \tag{2.1}$$

$$NormalizedAngularRMSD = \frac{1}{1 + \left(\frac{AngularRMSD}{\pi/4}\right)^2} \tag{2.2}$$

where x_{1i} is the vector of / or w angle sequence for n residues predicted from the amino acid sequence and x_{2i} is the vector of the corresponding observed ϕ or ψ angle sequence for n residues in the model.

ROSETTA centroid energy terms

We use 12 ROSETTA [76, 77] centroid energy terms as features that include residue environment (env), residue pair interactions (pair), cβ density (cbeta), steric repulsion (vdw), radius of gyration (rg), packing (cenpack), contact order (co), statistical potentials for SS formation (hs_pair, ss_pair, sheet, rsigma), and centroid hydrogen bonding (cen_hb). We apply sigmoidal transformations to scale the energy terms before using them as features.

Sliding window

In order to capture the local interactions among residues, we employ a sliding window of 21 around the central residue (i.e., 10 residues on both sides) similar to [55]. This results in a 483-dimensional feature vector for each residue. N- or C-terminal residues having one or more missing neighbors on either side are padded with additional 0's to match the feature dimension.

2.3.3 Architecture of stacked deep ResNets

Figure 2C shows a high-level overview of the architecture of our stacked deep ResNets. Each network consists of 13 residual blocks with three 1-dimensional convolutional layers in each block. We adopt the bottleneck design [36] for the residual modules with a kernel size of 1 x 1, 1 x 3, and 1 x 1, respectively, for three convolutional layers in each residual block. Here, the 1 x 1 layers at the beginning and at the end of each residual block are used to reduce and restore the dimensionality of the feature vector, thus maintaining consistency in the dimensionality of feature maps throughout the network. We input L x 483 x 1 feature to a convolutional layer with a kernel size of 1 x 7 and filter of 64 x 1. Afterwards, the feature is transformed into 64-dimensional feature vector using a 1 x 1 convolutional layer of the first residual block with a filter size of 64 x 1. In each of the residual blocks, a

'shortcut connection' between the input and the output layer, skipping the intermediate layer is established. This connection works as identity mapping and its outputs are added to the output from all previously stacked layers and passed to the pre-activation phase of the final layer of a residual block. As depicted in Figure 2C, the input x in the i^{th} layer is added to the input of the final layer of a residual block. Therefore, the activation in the output layer for a specific residual block is applied to the $x_i + F(x_{i+1})$

$$y = F(x_{i+1}, \{W_i + 1\} + W_s X_i) \tag{2.3}$$

where **F** is the ReLU activation function, W is the weight vector in a particular layer i, W_s is the additional parameter to the model, representing the linear projection by the shortcut connection, applied to match the dimension, which is implemented by 1 x 1 convolution. Our entire deep residual network is divided into three stages with three residual blocks in the first stage, four in the second stage, and six in the last stage. In Figure 2C, n defines the output channels for the residual block in each stage that are set to 128, 256, and 512, respectively. Therefore, in the first block of each stage, the feature map is halved and the filter size is increased by a factor of 2. The dimensionality of the feature map remains the same in the consecutive blocks in a stage. We apply batch normalization on the input features before passing to the convolutional layer in the first stage of the residual network. The utilization of this setting, therefore, minimizes the internal covariate shift as well as the need for the Dropout [78]. At the end of the residual blocks, we use an average pooling layer with a pool size of 2 that reduces the number of parameters and helps in faster computation. Finally, a flatten layer accepts the pooled feature map to transform into a 1-dimensional vector and passes to the fully connected (i.e., dense) layer.

2.3.4 Model training

We collect submitted models from CASP9 and CASP10 [79, 80] experiments for a total of 220 protein targets, whose experimental structures are publicly available. On average, there are 282 models per target. To remove redundancy, we perform MUFOLD clustering [81] and select the centroid of the top 10 clusters. It should be noted that not all the targets have all 10 clusters and MUFOLD fails to execute for 5 targets, resulting in a total of 2,130 redundancy-removed models having 303,675 residues for 215 CASP targets. We prepare four sets of features at 1, 2, 4, and 8Å error thresholds with the same training data by assigning a binary label to each one of the residues after calculating the errors between the Ca atoms of each of the residues in the model and the corresponding aligned residue in the experimental structure using the LGA program [67]. We assign a label of 1 (positive class) to the feature set of each of the residues if the error is within rÅ (where r ϵ {1, 2, 4, 8}Å following GDT-TS), 0 (negative class) otherwise. We train an ensemble of four independent ResNet models using the four sets of features at 1, 2, 4, and 8Å error thresholds. As Conv1D accepts an input shape of a 3D tensor with batch, steps, and channel, respectively, we reshape the feature vector into L x 483 x 1 prior to passing to the input layer. We train the networks with the maximum number of epochs of 120 and an optimal batch size of 64 that best fits the GPU limit. Also, to avoid overfitting, we use EarlyStopping callback of Keras [82] with a patience value of 20. We optimize the model using the binary crossentropy loss function and first-order gradient-based Adam optimizer [83].

2.3.5 Residue-level ensemble error classifications and their combination for model quality estimation

Figure 2D shows the residue-level ensemble classifications and their combination for model quality estimation. We consider each one of the four deep ResNet models as an independent

residue-specific error classifier, while their ensemble collectively estimates the structural quality of a model. For each classifier, the output layer with the sigmoid activation function predicts the likelihood of residue-level errors at 1, 2, 4, and 8Å error thresholds. We set a likelihood cutoff of 0.5 to convert the likelihood of a residue-level error to binary classification, where a likelihood value greater than the cutoff is classified as 1, indicating the residue-specific error to be within rÅ error level (ϵ {1, 2, 4, 8}Å) and 0 otherwise. For a given error threshold at rÅ, we calculate the aggregated error for the whole model by summing up the number of residues belonging to the positive class N r. Analogous to GDT-TS score, we estimate the quality of a model by combining the ensemble of residue-level classifiers as:

$$QDeep - score = \frac{N1 + N2 + N4 + N8}{4L} \tag{2.4}$$

where N1, N2, N4, and N8 are the number of aligned residues within 1, 2, 4, and 8Åerror thresholds, respectively, and L is the length of the target protein. Consequently, QDeep-score lies between [0, 1] with a higher value indicating better quality.

2.3.6 Evaluation method and programs to compare

We validate the individual residue-level classifiers using the 'stage 2' model pool (150 models/target) for 82 CASP11 targets[84] with publicly available experimental structures. For evaluating model quality estimation performance, we use the stage 2 model pool for 40 and 20 targets from CASP12 and CASP13, respectively [59, 84], with a total of 9,000 models for both datasets. The training and test datasets are non-overlapping with an average pairwise sequence identity of 21%. We use three evaluation criteria to measure the performance of model quality estimation: (i) ability to reproduce the true model-native similarity scores, (ii) the ability to find the best model, and (iii) the ability to distinguish between good and bad models. For the first criterion, we use average per-target and global Pearson, Spearman,

and Kendall's Tau correlations between the estimated scores and the true GDT-TS accuracy considering all models in a given dataset. Consequently, a higher correlation indicates better performance. For the second criterion, we use average GDT-TS loss which is the difference between the true GDT-TS of the top model selected by the estimated score and that of the most accurate model in the pool. A lower loss, therefore, indicates better performance. For the third criterion, we perform receiver operating characteristics (ROC) analysis using a cutoff of GDT-TS = 0.4 to separate good and bad models. Meanwhile, the area under ROC curves (AUC) quantifies the ability of a method to distinguish good and bad models. We compare our new distance-based single-model method QDeep with state-of-the-art single-model quality estimation methods that include ProQ2 [49], ProQ3 [55], ProQ3D [14], ProQ4 [32, 59], 3DCNN [31], MESHI [85], and VoroMQA [86]. For CASP12 targets, all tested methods are run locally with parameters set according to their respective papers. For CASP13, we directly obtain quality estimation predictions submitted by the tested methods from the data archive of the CASP official website.

2.4 Results

2.4.1 Validation of individual residue-level classifiers

We validate the performance of our individual deep ResNet-based classifiers on 82 CASP11 targets. Figure 2.2 shows the ROC curves for each of the classifiers trained at error thresholds of 1, 2, 4, and 8Å, respectively. All individual classifiers achieve AUC values ∼0.8, demonstrating their effectiveness at various error thresholds. Of note, the AUC values steadily increase at higher values of error thresholds. This is not surprising because at a lower error threshold, the proportion of residues belonging to the positive class is very low. That is, the number of positive and negative labels is extremely unbalanced. For example, in the training dataset, the ratios between the positive and negative labels are 0.27 (88,36/331,318),

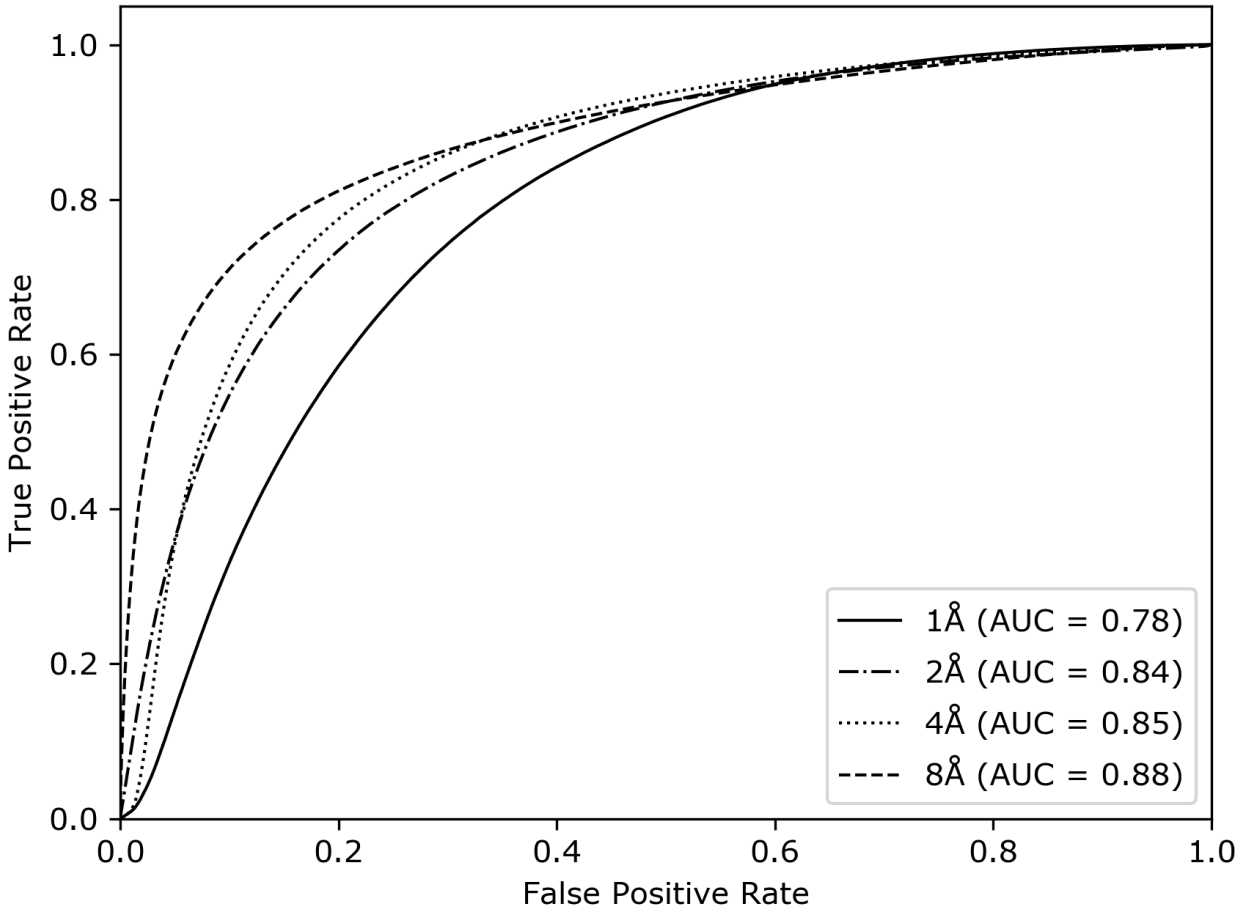

Figure 2.2: Accuracy of the individual residue-level classifiers at 1, 2, 4, and 8Å error thresholds on the validation set of 82 CASP11 targets

0.54 (147,194/273,084), 0.93 (202,373/217,927), and 1.57 (256,740/ 163,566) for 1, 2, 4, and 8Å error thresholds, respectively. The sparsity of positive labels at lower error thresholds may be the reason behind their somewhat lower performance. Nonetheless, they still deliver reasonable residue-level classification performance. Binary classification performance metrics, such as F1, MCC, Precision, and Recall for the individual classifiers at various error thresholds are reported in Supplementary Table 2.4.

Table 2.1: Performance of single-model quality estimation methods on CASP12 and CASP13 stage 2 datasets, sorted in decreasing order of average per-target Pearson correlations.

Dataset	Method	Avg r[a]	Avg ρ[b]	Avg τ[c]	Avg loss[d]	Global r[e]	Global ρ[f]	Global τ[g]
CASP12	QDeep	**0.740**	**0.657**	**0.492**	**0.051**	**0.863**	**0.871**	**0.678**
	ProQ3D	0.688	0.631	0.467	0.086	0.851	0.847	0.660
	3DCNN	0.661	0.585	0.427	0.081	0.834	0.818	0.620
	ProQ2	0.624	0.556	0.404	0.091	0.784	0.770	0.577
	ProQ3	0.604	0.536	0.390	0.071	0.806	0.793	0.600
	VoroMQA	0.560	0.502	0.362	0.105	0.604	0.603	0.444
CASP13	QDeep	**0.752**	**0.692**	**0.512**	0.088	**0.866**	**0.868**	**0.678**
	ProQ4	0.733	0.667	0.507	0.089	0.667	0.642	0.491
	MESHI	0.713	0.663	0.492	**0.070**	0.833	0.845	0.659
	ProQ3D	0.671	0.619	0.457	0.084	0.849	0.811	0.626
	VoroMQA-A	0.665	0.606	0.442	0.092	0.769	0.767	0.574
	VoroMQA-B	0.651	0.592	0.429	0.072	0.754	0.750	0.554

Note: Values in bold represent the best performance.
[a,b,c]Per-target average Pearson, Spearman, and Kendall's Tau correlation with respect to true GDT-TS score
[d]Per-target average loss with respect to true GDT-TS score
[e,f,g]Global Pearson, Spearman, and Kendall's Tau correlation with respect to true GDT-TS score

2.4.2 Performance evaluation on CASP datasets

Table 2.1 reports the performance of our new method QDeep and five other top-performing single-model quality estimation methods on CASP12 and CASP13 stage 2 datasets. QDeep consistently outperforms all other tested methods for both CASP12 and CASP13 sets across almost all performance criteria. For instance, QDeep attains the highest per-target average Pearson correlation of 0.740 in CASP12, which is much better than the second-best ProQ3D (0.688). Additionally, QDeep attains the lowest average GDT-TS loss of 0.051, which is significantly lower than the second-best ProQ3 (0.071). Furthermore, QDeep always delivers the highest global correlations in CASP12. The same trend continues for CASP13 set, in which QDeep attains the highest per-target average Pearson correlation of 0.752, better than the second-best ProQ4 (0.733). In terms of GDT-TS loss in CASP13, however, MESHI attains the lowest average GDT-TS loss (0.070) as it adopts an additional loss enrichment step in its pipeline. QDeep has an unusually high GDTTS loss of 0.455 for the CASP13 target T1008, which raises its average GDT-TS loss. On further inspection, we find that the alignment depth of the MSA for T1008 is zero having no identifiable homologous sequences. If T1008 is excluded, the average loss of QDeep becomes 0.068. Considering all targets though, the average GDT-TS loss of QDeep in CASP13 is still comparable to the other tested methods. The global correlations attained by QDeep are always the highest in CASP13. Of note, ProQ4, the second-best performing method in CASP13 after QDeep in terms of per-target average correlations, consistently exhibit poor global correlations. MESHI, the method attaining the best average GDT-TS loss in CASP13 does not deliver top performance in terms of per-target average correlations. That is, there are complementary aspects of model quality estimation that can lead to performance trade-offs. Our new method QDeep strikes an ideal balance to deliver top-notch performance across various facets of model quality estimation simultaneously.

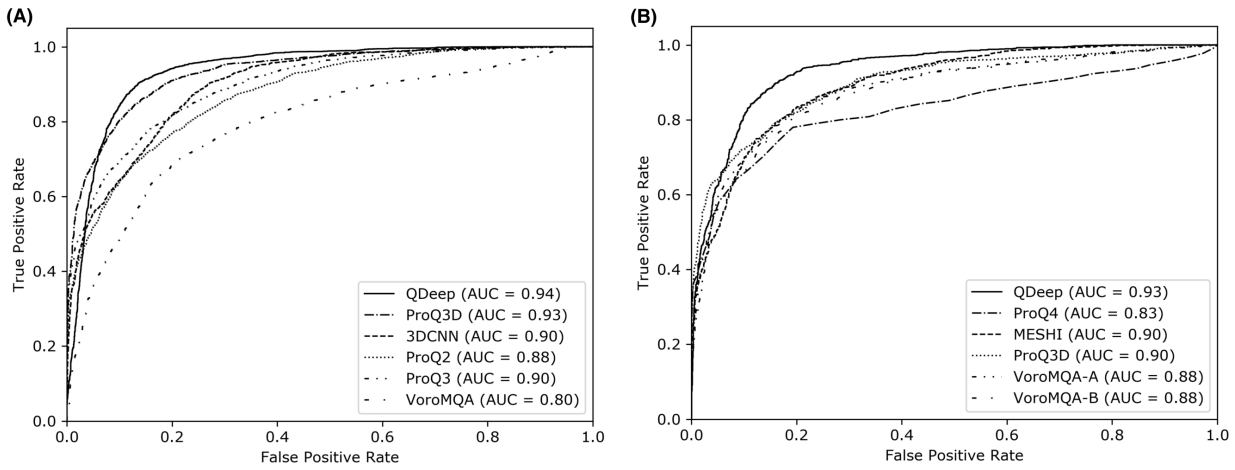

Figure 2.3: The ability of single-model quality estimation methods to distinguish good and bad models in (A) CASP12 and (B) CASP13 stage 2 datasets. A cutoff of GDT-TS = 0.4 is used to separate good and bad models

To investigate the ability of QDeep to distinguish good and bad models in comparison with the other tested methods, we perform ROC analysis using all models for all targets in CASP12 and again, QDeep consistently achieves the highest AUC values for both CASP12 and CASP13. The AUC of QDeep is slightly higher than the second-best ProQ3D in CASP12 and noticeably higher in CASP13, demonstrating its better performance in separating good and bad models compared to the others. It is interesting to note that among the other tested methods, deep learning-based approaches, such as ProQ4, ProQ3D, and 3DCNN routinely delivers better performance in both CASP12 and CASP13 datasets. Among the ProQ series of predictors, deep learning-based methods, such as ProQ3D and ProQ4 perform better than SVM-based approaches like ProQ2 and ProQ3. While these methods themselves are intrinsically different making it difficult if not impossible to firmly conclude the underlying cause of their performance differences, the trend of superior performance of deep learning-based methods may indicate the inherent advantage of transitioning from traditional machine learning to deep learningbased approaches for model quality estimation task.

To investigate the effect of various deep architectures on quality estimation performance

Table 2.2: Performance comparison of deep ResNet models used in QDeep with other deep learning architectures on CASP12 and CASP13 stage 2 datasets.

	CASP12 stage 2				CASP13 stage 2			
	Avg r[a]	Avg ρ[b]	Avg τ[c]	Avg loss[d]	Avg r[a]	Avg ρ[b]	Avg τ[c]	Avg loss[d]
ResNet	**0.740**	**0.657**	**0.492**	**0.051**	**0.752**	**0.692**	**0.512**	0.088
LSTM	0.716	0.596	0.452	0.059	0.735	0.668	0.500	0.116
CNN	0.657	0.581	0.433	0.097	0.735	0.660	0.487	0.116

Note: Values in bold represent the best performance.
[a,b,c]Per-target average Pearson, Spearman, and Kendall's Tau correlation with respect to true GDT-TS score
[d]Per-target average loss with respect to true GDT-TS score

when everything else remains the same, we compare deep ResNet with long short-term memory (LSTM) [87] and CNN [88] by performing controlled experiments. Equivalent to the ResNet model ensemble employed in QDeep, we train ensembles of four independent LSTM and CNN residue-specific error classifiers at 1, 2, 4, and 8A$^\circ$ error thresholds using the same features sets and same training data used in QDeep. The model architectures and training procedure for the LSTMs and CNNs are described in Supplementary Methods. Table 2.2 presents the head-to-head performance comparison between ResNets, LSTMs, and CNNs on CASP12 and CASP13 stage 2 datasets. The results show that ResNet delivers the best performance across all performance criteria, while LSTM consistently outperforms CNN. ResNet attains the highest per-target average correlations and the lowest average GDT-TS losses in both CASP12 and CASP13 sets. LSTM attains an average GDT-TS loss of 0.059, which is lower than all tested methods in CASP12 and second only to the ResNet architecture of QDeep. In CASP13, LSTM delivers better per-target average Pearson (0.735) and Spearman (0.668) correlations than all other methods except ResNet-based QDeep. The results further emphasize the advantage of using sophisticated deep architecture, such as LSTM for model quality estimation. Our new method QDeep goes one step further by training an ensemble of state-of-the-art stacked deep ResNets classifiers, delivering the best predictive performance. Meanwhile, the novel use of distance information in QDeep

substantially improves the performance, as discussed later. In summary, the advantage of QDeep in single-model quality estimation over the others is manifold.

2.4.3 Impact of deeper sequence alignment

Table 2.3: Performance comparison of variants of QDeep on CASP12 and CASP13 stage 2 datasets.

	CASP12 stage 2				CASP13 stage 2			
	Avg. r[a]	Avg. ρ[b]	Avg. τ[c]	Avg. loss[d]	Avg. r[a]	Avg. ρ[b]	Avg. τ[c]	Avg. loss[d]
QDeep	0.740	0.657	0.492	0.051	0.752	0.692	0.512	0.088
QDeep DeepMSA	0.741	0.667	0.505	0.062	0.777	0.720	0.538	0.084
QDeep NoDistance	0.677	0.601	0.442	0.065	0.668	0.613	0.445	0.091

Since a large number of features used in QDeep is dependent on MSA, which has been shown to significantly affect contact prediction, SS prediction and threading [72], we replace our alignment generation component with DeepMSA [72] to generate deeper sequence alignments by integrating whole-genome and metagenome sequence databases and retrain the ensemble of four stacked deep ResNet models with the same architecture and training procedure mentioned earlier (hereafter called QDeep^{DeepMSA}). To study the impact of deeper MSA in model quality estimation, we perform head-to-head performance comparison between QDeep and QDeep^{DeepMSA}. To make a fair comparison, we use the same test datasets of CASP12 and CASP13 with the same feature sets. As reported in Table 2.3, QDeep^{DeepMSA}further improves per-target average correlations in both datasets. The improvement is particularly noticeable for CASP13, in which QDeep^{DeepMSA} attains average per-target Pearson, Spearman, and Kendall's Tau correlations of 0.777, 0.720, and 0.538, respectively, that are much higher than QDeep trained on standard MSA. In terms of average GDT-TS loss, QDeep^{DeepMSA} results in slight improvement in CASP13 but visible worsening in CASP12. In Supplementary Figure 2.4, we show the performance of the individual classifiers at 1, 2,

4, and 8Å error thresholds by performing ROC analysis on our validation set comprising of 82 CASP11 targets. Except 8Å error threshold, all other deep ResNet classifiers trained on deeper alignments attain higher AUC values compared to their counterparts using standard MSA. That is, deeper sequence alignments can be advantageous to further improve the performance of our new quality estimation method QDeep, particularly by enhancing its ability to better reproduce true model-native similarity scores. To understand whether it is possible to further improve the performance by fine-tuning the hyperparameters of the ResNet architecture to better leverage the deeper sequence alignments, we train shallower and deeper ResNet architectures. For the shallower architecture, we stack 2 blocks in each of the three stages, resulting in a total of 6 residual blocks for each independent residue-level error classifier. The deeper architecture consists of a total of 20 residual blocks for each independent residue-level error classifier having 6, 7, and 7 blocks sequentially for the three stages. As described in Supplementary Methods, these two variants of our original 13-residual-blocks-QDeep^{DeepMSA} are trained using the same features with deep MSAs and same training data. Head-to-head performance comparison reported in Supplementary Table 2.5 reveals minor performance variations between the variants, with all architectures outperforming the other tested methods for both CASP12 and CASP13 sets in majority of performance criteria. The deeper architecture with 20 residual blocks performs better than the shallower architecture with 6 residual blocks, particularly for CASP13. Our original ResNet architecture of QDeep^{DeepMSA} consistently delivers the best performance over the variants across all assessment metrics, validating its effectiveness for quality estimation

2.4.4 Contribution of distance information

To evaluate the contribution of distance information in QDeep, we retrain an extra set of the same ensemble classifiers at 1, 2, 4, and 8Å error thresholds after excluding distance-based features, while still utilizing deep MSA for feature generation (hereafter called QDeep^{NoDistance}).

Head-to-head performance comparisons on the same test sets of CASP12 and CASP13 datasets reveal that QDeep^{NoDistance} performs much worse than the original distance-based QDeep method, let alone its variant QDeep^{DeepMSA} trained using deep alignments and distance information. As shown in Table 2.3, QDeep^{NoDistance} substantially degrades per-target average Pearson, Spearman, and Kendall's Tau correlations in both CASP12 and CASP13 sets. There is also a noticeable increase in the average GDT-TS loss. Clearly, the exclusion of distance information negatively affects the estimation of model quality performance. The results underscore the importance of incorporating distance information in single-model quality estimation methods, such as QDeep.

2.5 Conclusion

This article presents QDeep, a new distance-based single-model protein quality estimation method based on residue-level ensemble error classifications using stacked deep ResNets. Experimental results show that QDeep works much better than existing approaches across various accuracy measures of model quality estimation. QDeep outperforms not only currently popular ProQ series of methods including its most recent editions ProQ3D and ProQ4 but also top-ranked single-model quality estimation methods participating in the most recent 13th edition of CASP. Among the competing methods, deep learning-based approaches show a general trend of superior performance. Our new method QDeep takes a leap forward by employing cutting-edge deep learning architecture to effectively integrate predicted distance information with other sequential and structural features, leading to improved performance. Different from the other state-of-the-art methods, such as ProQ4 and MESHI that focus only on some aspects of quality estimation, our method works well on a wide-range of accuracy metrics to deliver an overall well-rounded performance. Controlled experiment on multiple datasets confirms that the improved performance of QDeep is primarily attributed to our effective integration of distance information that can be further improved in part by

incorporating deeper sequence alignments. This should make distance-based protein model quality estimation a promising new avenue for many more single-model methods in the near future.

2.6 Acknowledgements

This work was made possible in part by a grant of high performance computing resources and technical support from the Alabama Supercomputer Authority.

2.7 Funding

This work is partially supported by National Science Foundation (NSF) grant IIS-2030722 and NSF CAREER award DBI-1942692 (to D.B.).

2.8 Supplementary Information

Table 2.4: Performance of individual classifiers at 1, 2, 4, and 8Å error thresholds on 82 CASP11 targets.

Dataset	r[a]	F[b]	MCC[c]	Precision	Recall
CASP11	1Å	0.49	0.33	0.51	0.48
	2Å	0.70	0.52	0.74	0.66
	4Å	0.81	0.57	0.76	0.85
	8Å	0.85	0.56	0.83	0.88

[a]Deep ResNet classifier trained at a distance threshold of r
[b]F1 score
[c]Matthew's correlation coefficient

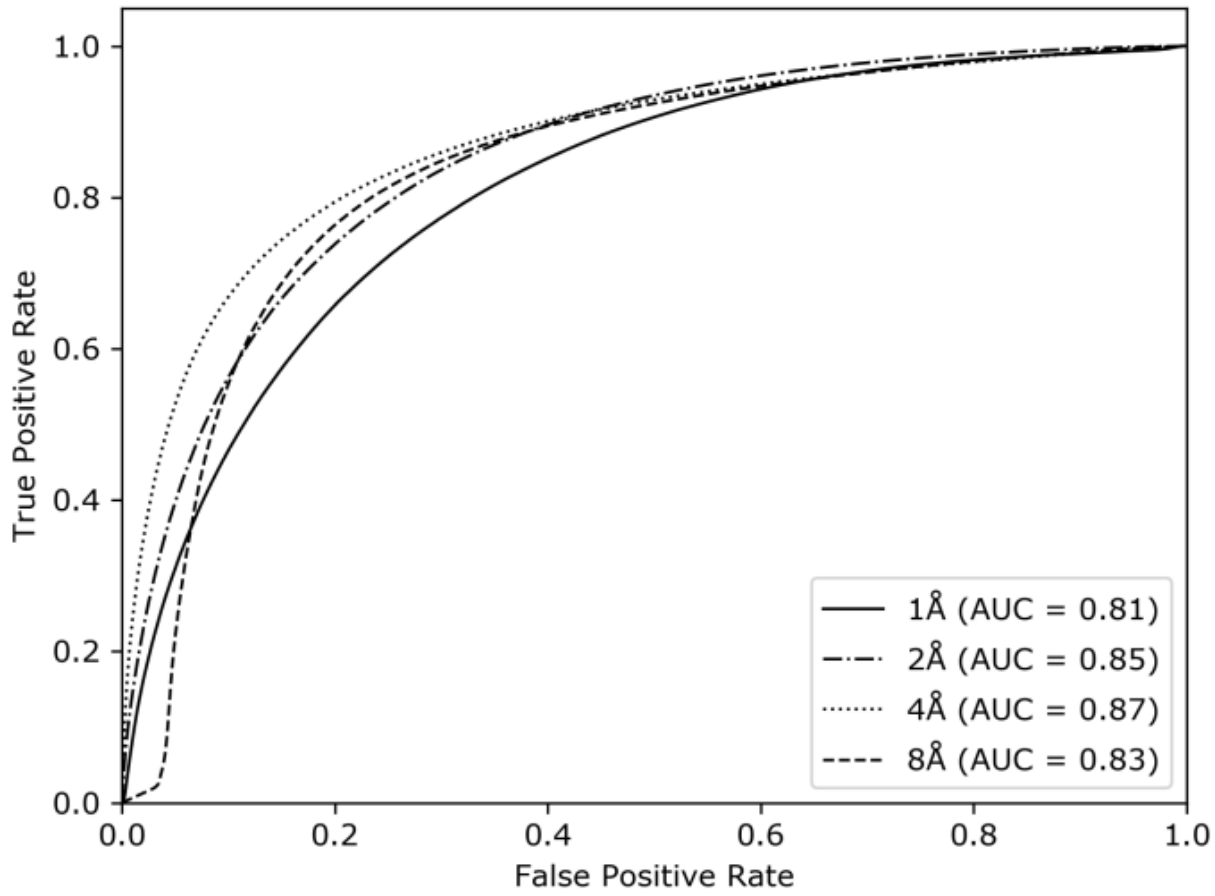

Figure 2.4: Accuracy of the individual residue-level classifier at 1, 2, 4 and 8Å error thresholds on 82 CASP11 targets. The classifiers are trained using the features generated by integrating deep MSA

2.8.1 Supplementary Method

Architectures of various deep learning models for model quality estimation

To investigate the effect of various deep learning models on quality estimation performance when everything else remains the same, we train Long Short-Term Memory [87] and Convolutional Neural Networks [88] in addition to the deep ResNet architecture used in QDeep.

Long Short-Term Memory (LSTM):

We construct an LSTM network by stacking LSTM layer with 64 LSTM blocks and two subsequent hidden dense layers with 32 and 16 nodes, respectively. Similar to the deep ResNet model ensemble employed in QDeep, we train ensembles of four independent LSTM residue-specific error classifiers at 1, 2, 4, and 8Å error thresholds using the same features sets and same training data used in QDeep. We use "relu" activation function for each of the dense layers, and "sigmoid" activation function for the output layer to return a probability value between 0 and 1 for each of the classes by using a probability threshold of 0.5 for each of the predictors to classify the residue-level error to be within rÅ (where r ϵ {1, 2, 4, 8}Å).

Convolutional Neural Network (CNN):

We construct a CNN architecture having 5 sequential stacked layers. We use 64 filters for each of the layers with a kernel size of 3 and "relu" activation function. To reduce the feature dimensionality, we add a MaxPooling1D layer with a pool size of 2 after every two convolutional layers. Additionally, to scale the activation we use BatchNormalization layer after every 2 convolutional layers. To reduce overfitting, we add a Dropout layer with a 50% dropout after every 2 convolutional layers. Subsequently, we add a Flatten layer followed by a fully connected dense layer as an output layer. The Flatten layer transforms the feature into 1D vector before it is passed to the output layer. Once again, we train ensembles of four independent CNN residue-specific error classifiers at 1, 2, 4, and 8Å error thresholds using the same features sets and same training data used in QDeep. We use "sigmoid" activation function in the dense layer to return a probability value between 0 and 1 for each of the classes by using a probability threshold of 0.5 for each of the predictors to classify the residue-level error to be within rÅ (where r ϵ {1, 2, 4, 8}Å).

Training various architectures of deep residual neural networks (ResNets)

In addition to the original deep ResNet model employed in the QDeep method that consists of 13 residual blocks (hereafter called $ResNet^{B13}$), we investigate the effect of the hyperparameters of the ResNet architecture on quality estimation performance by training additional ResNet models. We vary the number of residual blocks to train two variants of the original ResNet architecture: one shallower ResNet having 6 residual blocks for each independent residue-level error classifier (hereafter called $ResNet^{B6}$) and the other deeper ResNet having 20 residual blocks for each independent residue-level error classifier (hereafter called $ResNetB^{20}$). For ResNetB6, we stack 2 blocks in each of the three stages, whereas for $ResNetB^{20}$, we stack 6, 7 and 7 blocks sequentially for the three stages. Similar to the original $ResNetB^{13}$ architecture, we adopt the same bottleneck design [36] for $ResNetB^{6}$ and $ResNetB^{20}$) architectures. We use the same set of features powered by deep multiple sequence alignments and the same training data for training. For the additional ResNet architectures of $ResNetB^{6}$ and $ResNetB^{20}$, we independently train ensemble of four stacked deep ResNet models for residue-level error classifications at 1, 2, 4, and 8Å error thresholds using the same training method as used for our original $ResNetB^{13}$ (QDeep) architecture.

Table 2.5: Performance comparison of various ResNet architectures on CASP12 and CASP13 stage 2 datasets. Values in bold represent the best performance.

	CASP12 stage 2				CASP13 stage 2			
	Avg.r[a]	Avg.ρ[b]	Avg.τ[c]	Avg. loss[d]	Avg.r[c]	Avg.ρ[c]	Avg.τ[c]	Avg. loss[d]
$ResNetB^{6}$	0.722	0.640	0.482	0.070	0.710	0.656	0.484	0.109
$ResNetB^{13}$ (QDeep)	0.741	0.667	0.505	0.062	0.777	0.720	0.538	0.084
$ResNetB^{20}$	0.716	0.637	0.478	0.065	0.742	0.684	0.507	0.103

[a,b,c]Per-target average Pearson, Spearman and Kendall's Tau correlation with respect to true GDT-TS score.
[d] Per-target average loss with respect to true GDT-TS score.

Chapter 3

iQDeep: an integrated web server for protein scoring using multiscale deep learning models

3.1 Abstract

The remarkable recent advances in protein structure prediction have enabled computational modeling of protein structures with considerably higher accuracy than ever before. While state-of-the-art structure prediction methods provide self-assessment confidence scores of their own predictions, independent and open access system for protein scoring is still needed that can be applied for a broad range of predictive modeling scenarios. Here, we present iQDeep, an integrated and highly customizable web server for protein scoring, freely available at http://fusion.cs.vt.edu/iQDeep. The underlying method of iQDeep employs multiscale deep residual neural networks (ResNets) to perform residue-level error classifications, and then probabilistically combines the error classifications for protein scoring. By adjusting the error resolutions, our method can reliably estimate the standard- or high-accuracy variants of the Global Distance Test metric for versatile protein scoring. The performance of the method has been extensively tested and compared against the state-of-the-art approaches in multiple rounds of Critical Assessment of Techniques for Protein Structure Prediction (CASP) experiments including benchmark assessment in CASP12 and

CASP13 as well as blind evaluation in CASP14. The iQDeep web server offers a number of convenient features, including (i) the choice of individual and batch processing modes; (ii) an interactive and privacy-preserving web interface for automated job submission, tracking, and results retrieval; (iii) web-based quantitative and visual analyses of the results including overall estimated score and its residue-wise breakdown along with agreements between various sequence- and structural-level features; (iv) extensive help information on job submission and results interpretation via web-based tutorial and help tooltips.

Keywords: Protein scoring; Accuracy estimation; Deep learning; Protein structure prediction

3.2 Introduction

Protein scoring is a critical component of protein structure prediction [13, 21, 24, 84, 89]. Protein scoring has been gaining noticeable attention in the Critical Assessment of Protein Structure Prediction (CASP) experiments under the accuracy estimation category [62, 90, 91, 92]. Promising progress has been made in the CASP14 accuracy estimation category [90] with various deep learning-based methods performing well. Among them is our previously published method QDeep [21], introducing several new advances for the first time including the incorporation of the deep residual neural networks (ResNets) architectures for protein scoring, effective integration of predicted inter-residue interaction with other sequential and structural features, and the use of ensemble learning. With rapid new developments in the field of protein structure prediction [2, 9, 93], however, the scoring resolution used in our original QDeep method no longer represents the state of the art. Recent advances in deep learning-based protein structure prediction methods such as AlphaFold2 [2] have enabled computational prediction of protein with considerably higher accuracy than what has been achieved before. As such, the development of high-resolution protein scoring methods commensurate with the increasing accuracy of structure prediction methods is of critical importance.

While our original QDeep method uses the Global Distance Test Total Score (GDT-TS) [67] as the ground truth metric, the recent advances in structure prediction [9, 18, 67, 93] and refinement [20, 23, 94, 95, 96] make the high-accuracy variant of the Global Distance Test (GDT-HA) [97] more suitable as the ground truth metric. Furthermore, an integrated protein scoring framework that can alternate between the standard and high accuracy variants of the Global Distance Test on-demand in order to control the scoring resolution, can enhance the versatility of protein scoring by covering a broad range of predictive modeling scenarios. Open availability of such a versatile method via a publicly accessible web server has the potential for broad dissemination and a field-wide impact.

Here we present iQDeep, an integrated and fully configurable web server for protein scoring using multiscale deep learning models. iQDeep employs multiscale deep residual neural networks (ResNets) to perform residue-level error classifications at multiple predefined error resolutions, and then probabilistically combines the predictions from the multiscale error classifiers for protein scoring. Building on the original QDeep method, we train a new set of ResNet classifiers to perform residue-level ensemble error classifications at finer-grained error resolutions explicitly targeting the GDT-HA metric. The residue-level error classifications from the multiscale ResNets can then be probabilistically combined for quantitating the accuracy of a predicted protein model. By adjusting the resolutions and probabilistic combination of the multiscale ResNets, our method can reliably estimate the standard- or high-accuracy variants of the Global Distance Test metric for protein scoring. The interactive and privacy-preserving web interface of iQDeep allows customizable job submission, tracking, and results retrieval with quantitative and visual analysis along with extensive help information on job processing and results interpretation. The performance of the underlying methods has been rigorously tested and compared against the state-of-the-art approaches in multiple rounds of CASP experiments including benchmark assessment in CASP12 and CASP13 and blind evaluation in CASP14. The iQDeep web server is freely available at http://fusion.cs.vt.edu/iQDeep.

3.3 Results and Discussion

Our previously published method QDeep [21] performed quite well on CASP12 and CASP13 benchmarking datasets. Additionally, it has been tested in a strict blind mode in CASP14 under the group name "Bhattacharya-QDeep". We use the CASP14 dataset to evaluate the performance of iQDeep and compare against 24 groups participating in CASP14 accuracy estimation category, including our own group "Bhattacharya-QDeep". The CASP14 benchmarking dataset consists of 10,494 models for 70 targets submitted by the CASP14 tertiary structure predictors. Given the progress made in protein structure prediction, we use the high-accuracy variant of the Global Distance Test (GDT-HA) as the ground truth metric for evaluation. For performance assessment, we use a twofold evaluation criteria: (1) ability to reproduce the ground truth scores, and (2) ability to distinguish acceptable from incorrect models.

3.3.1 Reproducing ground truth scores

To evaluate the ability to reproduce the ground truth scores, we calculate the global Pearson correlation (Pearson r) and the average absolute difference (ΔGDT-HA) between the estimated scores and the ground truth GDT-HA scores for all models in the CASP model pool. Meanwhile, higher Pearson r and lower ΔGDT-HA indicate enhanced ability to reproduce the ground truth scores. Figure 3.1 shows the performance of iQDeep and the groups participating in CASP14 accuracy estimation category, including our own group "Bhattacharya-QDeep" employing the original QDeep method. The results demonstrate that iQDeep achieves state-of-the-art performance. For example, iQDeep attains a Pearson r of 0.646, outperforming "Bhattacharya-QDeep", and better than most of the CASP14 predictors except "BAKER-ROSETTASERVER", "BAKER-experimental", "3DCNN_prof", and "RaptorX-QA" groups [12, 20, 31]. Remarkably, iQDeep attains the lowest ΔGDT-HA of

0.119, which is significantly better than second best performer "RaptorX-QA", let alone our previously published method QDeep implemented in "Bhattacharya-QDeep". It is interesting to note that "BAKER-ROSETTASERVER" and "BAKER-experimental", despite attaining high Pearson r, underperform in terms of ΔGDT-HA. In contrast, iQDeep and two CASP14 groups "RaptorX-QA" and "3DCNN_prof" deliver consistent performance. In summary, iQDeep exhibits excellent all-round ability to reproduce ground truth scores.

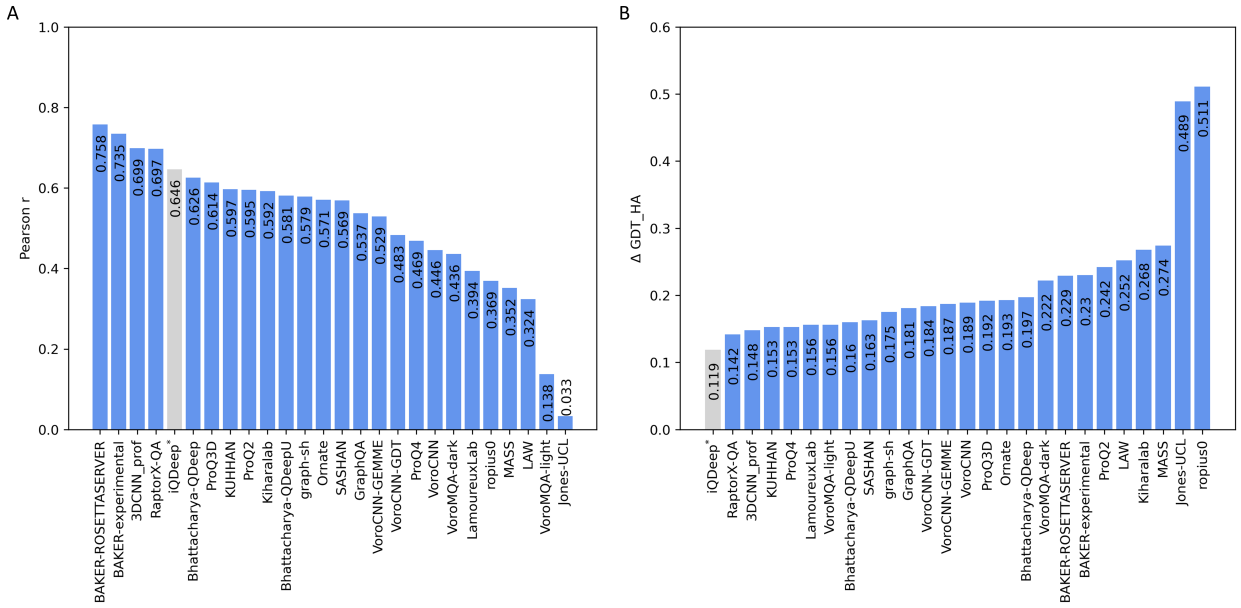

Figure 3.1: Reproducibility of ground truth scores for iQDeep and 24 groups participating in CASP14 accuracy estimation category in terms of (A) global Pearson correlation coefficient and (B) average absolute difference between the estimated scores and the ground truth GDT-HA scores in CASP14 dataset.* iQDeep is not a participating group in CASP14.

3.4 Distinguishing acceptable from incorrect models

In addition to reproducing the ground truth scores with high fidelity, the ability to discriminate between acceptable and non-acceptable prediction is critically important. We measure the separation between acceptable (GDT-HA ≥ 0.3 [98]) and incorrect models using a receiver operating characteristic (ROC) curve. As shown in Figure 5.2, the Area under the ROC Curve (AUROC) of iQDeep and our CASP predictor "Bhattacharya-QDeep" employing

the original QDeep method achieved AUROC of 0.826 and 0.816, respectively. Once again, iQDeep outperforms our previously published method QDeep. The AUROC of iQDeep is only slightly lower than the top performing CASP predictor "BAKER-ROSETTASERVER", and better than multiple well-known CASP groups including "ProQ2" [84, 99], "ProQ3" [62], "ProQ3D", and "VoroMQA" [13]. Overall, iQDeep can reliably discriminate between acceptable and incorrect predictions.

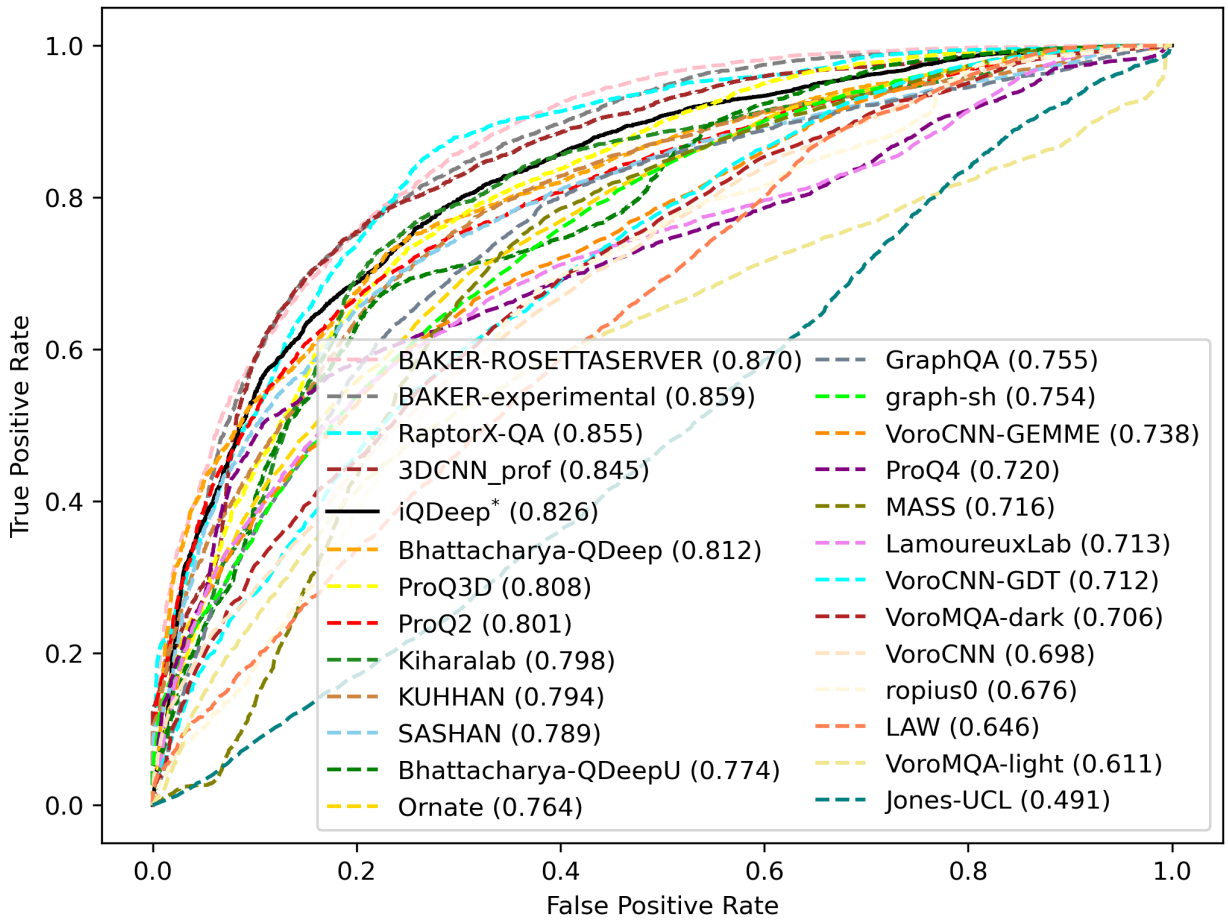

Figure 3.2: Distinguishability of acceptable vs. incorrect models for iQDeep and 24 groups participating in CASP14 accuracy estimation category using a receiver operating characteristic (ROC) curve. The numbers reported are the Area under the ROC Curve (AUROC) values.* iQDeep is not a participating group in CASP14.

3.5 Materials and Methods

3.5.1 Overview of iQDeep pipeline

Figure 3.3 shows the flowchart of iQDeep webserver consisting of front-end and back-end modules. The front-end module performs the input validation, allows users to select customizable job parameters, monitors the job processing, and enables results retrieval with quantitative and visual analysis. The back-end module processes the job, manages the job queue, and ensures web storage management and retrieval of job-related metadata. Users are able to protect the privacy of their jobs by selecting a private mode of job processing and results retrieval.

Front-end module

The front-end of the web server provides an interface for job submission, performs input validation, and returns the results back to the user. While the default job submission mode requires minimum user input that includes a valid protein structure and a job name, users can perform on-demand customization of the job parameters including selecting the scoring mode, scoring resolution, and job privacy. Users may select "single" or "batch" scoring mode for processing just a single or a pool of models, respectively. The scoring resolution parameter provides users the choice of selecting a specific set of multiscale deep learning models trained at error resolutions targeting the GDT-TS metric as implemented in our original QDeep work (scoring resolution "standard"), or the newly introduced finer-grained error resolutions explicitly targeting the GDT-HA metric (scoring resolution "high"). The privacy parameter allows the users to configure the privacy of their job. The input validation ensures correctness of all the inputs to ensure seamless job execution. The front-end module also offers a web-based analysis of the scoring results through easily interpretable and interactive web-based plots and three-dimensional molecular visualizations. An optional email address can also be

provided for automated status updates of the job via email.

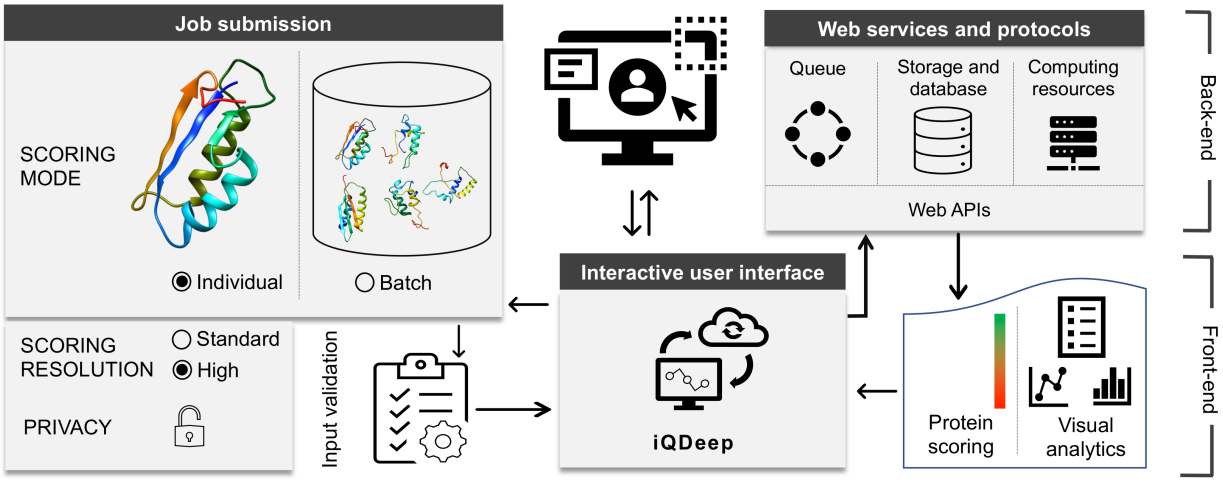

Figure 3.3: The flowchart of iQDeep web server for protein scoring consisting of the front-end module for automated job submission with input validation and interactive results analytics and the back-end module for processing and managing jobs.

Back-end module

The back-end module of iQDeep performs job management using the queuing layer, controls the database and web storages, processes protein scoring jobs in our in-house compute cluster, and maintains client-server communication. The back-end module actively communicates with the front-end module to accept validated jobs and return the results upon successful job completion. After a job is successfully submitted, the queuing layer of the back-end module starts preparing the job for processing using a first-in-first-out (FIFO) job scheduler. During the job processing, the queuing layer first stores the job-related information and parameters in the web storage and the corresponding job metadata in the database. It then sends the job to the computational node for processing based on the selected job parameters, while continuously monitoring the computational resource availability in our in-house compute cluster. The back-end module maintains the client-server protocol by establishing a two-way communication with the front-end module during the entire job processing phase to periodically send the job updates to the front-end. Upon completion of the job, the back-end

module sends the results back to the front-end for interactive and interpretable web-based visual and quantitative analytics, with an optional email notification sent to the user, if an email address is provided.

3.5.2 The architecture of multiscale deep learning models for protein scoring

iQDeep employs multiscale deep residual neural networks (ResNets) to perform multi-resolution protein scoring. The standard scoring resolution represents our original distance-based protein scoring method QDeep[21]. It employs an ensemble of four deep ResNets, each independently trained to classify the residue level errors at four different error thresholds of 1, 2, 4, and 8Å to target the GDT-TS [67] metric. The method leverages several distance-based alignment features, sequence profile information, consistency between the predicted and observed structural properties, and several biophysical energy terms, as described in the published article of QDeep [21]. The residue-level ensemble error estimates are then combined to estimate an accuracy score of a protein model, thereby estimating a probabilistic equivalent of the GDT-TS metric. Building on our prior work, in iQDeep we train a new set of ResNet classifiers having an identical architecture to perform residue-level ensemble error classifications at finer-grained error resolutions of 0.5, 1, 2, and 4Å to explicitly target the GDT-HA metric. The predictions from the finer-grained error classifiers can then be probabilistically combined for high-resolution protein scoring, as follows,

$$iQDeep\,score = \frac{N0.5 + N1 + N2 + N4}{4} \tag{3.1}$$

where N0.5, N1, N2, and N4 represent the fraction of residues estimated to be within 0.5, 1, 2, and 4Å from the corresponding residues in the experimental structure. The estimated score ranges between 0 and 1 with a higher score indicating better model accuracy.

3.5.3 Web server

Hardware and software

The iQDeep web server runs on a Linux x86_64 cluster with 2.50GHz Intel Xeon Silver 32-core processors. The underlying client-server architecture of iQDeep is implemented using server-side PHP and client-side Javascript scripting languages, and deployed through an Apache web server with Common Gateway Interface (CGI). iQDeep uses MySQL relational database management system (RDBMS) for storing, retrieving, and updating job-related information. Additionally, it uses the WebGL-based JavaScript library 3Dmol.js [100] for interactive 3D molecular visualization. The iQDeep pipeline for protein scoring is implemented using Python. The webserver is compatible with most modern web browsers including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge.

Input and output

The iQDeep web server allows users to submit a job using only two required fields including a job name and a valid protein structure for scoring. However, users can customize several job parameters including scoring mode, scoring resolutions, and job privacy. An optional email address can also be provided for automated status updates of the job via email. The web server offers interactive results update using easy-to-interpret quantitative and visual analytics. In addition to providing global estimated accuracy scores, the web server provides the residue-level local accuracy estimation to identify reliable and unreliable regions of an input protein model along with interactive residue-wise visualization of estimated local accuracies. A 3D representation of the protein structure with an option to download the corresponding structure file is also provided. Users can also analyze several structural properties through easily interpretable web-based graphical alignment between the observed and predicted structural properties including secondary structure, solvent accessibility, and

residue-residue contact maps at various distance thresholds. The full set of results, including the global and local accuracy scores and text files containing the additional analyses, can be downloaded as a compressed zipped archive.

3.5.4 Acknowledgments

This work was partially supported by the National Institute of General Medical Sciences [R35GM138146 to D.B.] and the National Science Foundation [DBI2208679 to D.B.].

3.6 Supplementary Information

3.6.1 Architecture and training of multiscale deep learning models in iQDeep

Dataset curation

In order to train the multiscale deep neural network architecture in iQDeep, we obtained publicly available predicted protein models for 220 targets from the CASP9 and CASP10 experiments [101, 102]. Each target has an average of 282 protein models. In order to eliminate any redundancy we use MUFOLD [81] to cluster all the models for each target to identify the redundant models. For each target, we select a maximum of the top 10 clusters and collect the model that represents the centroid of a corresponding cluster. As a result, we obtained a final training set of 2,130 models for all targets.

Features extraction

To generate features for each of the models in the training dataset, first, we execute several publicly available prediction programs to generate sequence-based descriptors as outlined

below:

1. Sequence-derived information:

   Multiple sequence alignment: We use HHblits [68] to generate Multiple Sequence Alignment (MSA) for each of the protein sequences in the training dataset. For HHblits, we use E-value inclusion of 10-3, query sequence coverage of 10%, and pairwise sequence identity of 90% to search Uniclust30 [69] database to generate MSAs.

   Position-Specific Scoring Matrix: We use PSI-BLAST v2.2.26 [74] with E-value of 0.001 for searching the NR database in order to generate Position-Specific Scoring Matrix (PSSM).

   Secondary structure, solvent accessibility, and backbone angles: We use SPIDER3 [103] to predict secondary structure, solvent accessibility, and backbone angles (ϕ, ψ). We transform the 8-state predicted secondary structure into three-state secondary structures of Helix, Strand, and Coil. Subsequently, we transform the predicted accessible surface area into 2-state solvent accessibility including buried and exposed. In addition to predicting the structural properties, we calculate the observed secondary structure, solvent accessibility, and backbone angles (ϕ, ψ) using the DSSP [104].

   Distance histogram: We use DMPfold [52] to predict the inter-residue distance probability distribution at predefined distance bounds. We obtain the rawdistpred.current file generated as the initial prediction without having any iterative refinement step. For each of the interacting residue pairs, there are 20 distance bins with associated likelihoods.

2. Featurization: For each of the residue in a protein model in the dataset, we generate 23 features as outlined below:

   Sequence conservation score: We generate 1 feature of normalized information per-position score. First, we extract the information per-position score from the predicted Position Specific Scoring Metrix (PSSM) and then apply a sigmoidal transformation

to generate a normalized score between 0 and 1 as follows:

$$\frac{1}{1+e^{-}x_i} \tag{3.2}$$

Where x_i represents the sequence conservation score in the PSSM of a residue i in the model.

The number of effective sequences: We generate 1 feature of the number of effective sequences (NEFF) using the generated MSA. It is calculated as the summation of the nth reciprocal of the total number of sequences in the MSA having a sequence identity greater than 80%. We use NEFF as a global feature for a protein model.

Secondary structure and solvent accessibility: We generate 2 features for each residue in the model for secondary structure and solvent accessibility by calculating the agreement between predicted and observed secondary structure and solvent accessibility. We use DSSP [104] to calculate the observed secondary structure and solvent accessibility of the protein model. We assign 1 if there is a match between the predicted and observed secondary structures and 0, otherwise. Additionally, we calculate the squared error between the predicted and observed solvent accessibility and normalized the error between 0 and 1 using sigmoidal transformation.

Angular RMSD: We calculate 2 features of normalized angular RMSD between the predicted and observed dihedral angles (ϕ, ψ) as follows:

$$AngularRMSD = \sqrt{\frac{1}{n}\sum_i (min(x_{oi} - x_{pi}, 2\pi - |x_{oi} - x_{pi}|))^2} \tag{3.3}$$

$$NormalizedAngularRMSD = \frac{1}{1+\left(\frac{AngularRMSD}{\pi/4}\right)^2} \tag{3.4}$$

Distance-based features: We generate 5 distance-based alignment features between

the predicted and observed inter-residue distances, generated at 5 distance intervals of 6Å, 8Å, 10Å, 12Å, and 14Å. First, we use the DMPfold predicted distance histogram to extract interresidue distance probability at the corresponding distance intervals. Additionally, we calculate the observed distance histogram at 6Å, 8Å, 10Å, 12Å, and 14Å from the model. We then perform dynamic programming-based alignment between the predicted and observed distance maps using eigen-decomposition, resulting in 5 distance alignment scores. Finally, we assign the experimentally selected weights of 0.10, 0.25, 0.30, 0.25, and 0.10 to the distance alignment at 6Å, 8Å, 10Å, 12Å, and 14Å, respectively to generate 5 distance-based features.

Residue-specific centroid energy scores by ROSETTA: We calculate 12 energy terms for each model using Rosetta [76, 77] by adopting the centroid representation of the model. We use the "ref2015" score to calculate the residue environment (env) score, residue pair interactions (pair) score, c density (cbeta), steric repulsion (vdw), the radius of gyration (rg), packing (cenpack) density, contact order (co), statistical potentials for secondary structure formation (hs_pair, ss_pair, sheet, rsigma), and centroid hydrogen bonding (cen_hb). Once again, we use the sigmoid function to normalize the scores between 0 and 1 and use them as features.

3. Dataset preparation for training multiscale deep learning models We construct a 1-dimensional feature vector for each residue of the corresponding model by concatenating the aforementioned 23 features. We repeat this process for L residues in the model, resulting in L $\times$ 23 feature matrix. We employ a sliding window of 21 (i.e., 10 residues on both sides) around the central residue, resulting in 483-dimensional features. Afterwards, we assign binary labels to the resulting representative features for each residue by calculating the distance error between the C atom of an individual residue and the corresponding aligned residue in the experimental structure after performing the sequencedependent analysis (SDA) between the model and the corresponding experimental structure using the LGA program [67].

For training the deep learning models in our standard-resolution variant of iQDeep, we generate four sets of features at thresholds of 1, 2, 4, and 8Å, and assign a label of 1 to a feature vector if the distance between the aligned residues in the model and the residue in the experimental structure is within the specified threshold. In our high-resolution variant of iQDeep, we generate an additional four sets of features at thresholds of 0.5, 1, 2, and 4Å following the similar strategy

4. The architecture of ResNet classifiers in iQDeep The underlying deep learning models in iQDeep consist of deep Residual Neural Network classifiers (ResNets). Specifically, we use a ResNet40 architecture for each classifier, consisting of a 39-layer deep 1-dimensional convolutional neural network and a fully connected dense layer. The 39 convolutional layers are grouped into 13 residual blocks having 3 convolutional neural in each block. We use a kernel size of 1×1, 3×3, and 1×1, respectively for the three consecutive convolutional layers in the block, representing a bottleneck design that helps in faster training [36]. Furthermore, a "shortcut connection" is established between the first and the last layer of a residual block, bypassing the intermediate layer, thereby performing an identity mapping by combining the initial input features xi with the transformed x_{i+1} feature from the intermediate layer. It is noteworthy to mention that the initial input features are passed through an additional bottleneck layer with kernel size 1×1 to match the dimension of the output layer of the corresponding residual block. Afterward, ReLU activation is applied to the combined features y, represented as,

$$y = F(x_{i+1}, \{W_i + 1\} + W_s X_i) \tag{3.5}$$

Where W represents the weight vector for specific layer i and Ws is the term used to represent the linear projection of the input feature xi to match the feature dimension. We further organize the 13 residual blocks into three consecutive stages with each stage

having a different configuration. We group three residual blocks in the first stage, four in the second stage, and six in the third stage. Additionally, we vary the size of the filter of the residual blocks at every stage. We set a filter size of 128 for the convolutional blocks for the first stage and increase the filter size by a factor size of 2 with a filter size being 256 and 512 for the convolutional layers of the second and third stages respectively. Convolutional neural networks use filters to extract features from the original input vector and such increment in the size of the filter in the deeper layers helps to extract more meaningful features. [105].

The ResNet model takes as input a feature vector of size of $L\times483\times1$ where L represents the number of residues in a protein model and 483 is the number of features for each corresponding residue (see Section 4.6.1). The input is first embedded into a 64-dimensional feature vector with a convolutional layer with a kernel size of 1×7 and a filter size of 64×1. Subsequently, the feature vector is transformed using batch normalization before it is passed to the first residual blocks. The use of batch normalization serves to accelerate the training process and mitigate the internal covariation shift, thus eliminating the need for Dropout operation [78].

After the feature vector is passed to the residual blocks, the input feature vector is transformed by the initial convolutional layer within the block, which utilizes a kernel size of 1×1 and afilter size of 128. The purpose of this convolutional layer is to reduce the dimensionality of the input feature vector. Specifically, it transforms the input feature vector to a 64×1 dimension, which is suitable for the subsequent convolution operation. The convolution operation is performed using the reduced dimension feature vector, and it produces an output feature vector of 128×1. To further enhance the feature vector, the output from each convolutional layer is passed through a batch normalization operation. We use an epsilon and momentum of $2 \times 10{-}4$ and 0.9, respectively for the batch normalization. After the batch normalization, the output is then passed through a rectified linear unit (ReLU) activation function, which adds

a non-linearity to the output, allowing the network to learn complex features. This transformed output is then passed to the next layer in the block.

In a particular stage, the feature vector is processed in a consistent manner by subsequent blocks. A shortcut connection is employed, in which the output feature vector from the previous block is concatenated with its input feature vector. This technique enables the training of deep neural networks by allowing information to propagate more easily through the network. This process is repeated for all subsequent stages, with variations in the number of residual blocks and the number of filters for the corresponding convolutional layer in the network. At the end of the residual blocks, we use an average pooling layer with a pool size of 2 that helps in faster computation. Afterwards, we add a flatten layer that accepts the pooled features and transformed them into a 1D vector. The fully connected (dense layer) layer at the end, then accepts the 1D vector and applies sigmoidal activation to return a probability value.

5. Training of multiscale deep learning models For each of the variants of iQDeep, we train an ensemble of four ResNet classifiers. For the standard-resolution variant, we train the ensemble of ResNet classifiers using the generated features at four multiscale thresholds of 1, 2, 4, and 8Å. On the other hand, for the highresolution resolution variant of iQDeep, we train an ensemble classifier with the features generated at finer-grained multiscale thresholds of 0.5, 1, 2, and 4Å.

   During the training of each multiscale ResNet classifier, we use ridge regularization ($L2$ regularization) with each of the convolutional layers in the network with a parameter set to 10−3. It helps to prevent overfitting during training [106]. we train each of the classifiers with a maximum epoch set to 120 and a batch size of 64 to effectively optimize the memory utilization of our GPUs. We optimize the model using binary CrossEntropy loss function and first-order gradient-based Adam optimizer [83]. During the training, we exploit the *EarlyStopping* callback mechanism of Keras that conditionally stops the model training in case of unimproved validation loss for

20 consecutive iterations, imperatively helping in avoiding model overfitting.

6. Protein scoring using iQDeep In iQDeep, individual ResNet classifiers are utilized to estimate the likelihood of a residue in the input protein model being within a predefined deviation from the corresponding residue in the experimental structure. The ensemble of ResNet classifiers, either for standard-resolution or high-resolution in iQDeep, are employed to evaluate a protein model.

High-resolution protein scoring

The high-resolution scoring module of iQDeep utilizes an ensemble of four ResNet classifiers that have been trained using error thresholds of rÅ, where r belongs to the set 0.5, 1, 2, 4Å. Each of the four ResNet classifiers independently estimates a probability value for every residue in the model. The probability values computed by the ResNet classifiers for the respective error thresholds of rÅ are subsequently transformed into binary classes, where a probability value greater than 0.5 results in a positive class and a probability value less than or equal to 0.5 results in a negative class. A weighted combination of the number of positive classes, indicating the number of residues belonging to a specific rÅ, is then performed to score the protein model as follows:

$$iQDeep\,score_{high-resolution} = \frac{N0.5 + N1 + N2 + N4}{4} \tag{3.6}$$

where $N0.5$, $N0.1$, $N0.2$, and $N0.4$ represent the fraction of residues estimated to be within 0.5, 1, 2, and 4Å from the corresponding residues in the experimental structure. The estimated $iQDeep\,score_{high-resolution}$ score is equivalent to the GDT-HA score, ranging between 0 and 1 with a higher score indicating better model accuracy.

Standard-resolution protein scoring The standard-resolution scoring module of iQDeep similarly utilizes an ensemble of four ResNet classifiers trained at rÅ (rϵ1, 2, 4, 8Å). Each of the four ResNet classifiers independently estimates the probability of

a residue to be within rÅ. These probabilities are then transformed to binary values (1 or 0) using a probability threshold of 0.5, with a value of 1 indicating that the residue belongs to the corresponding resolution. Finally, a weighted combination of the number of residues predicted to be within the 1, 2, 4, or 8Å thresholds is used to score the protein model as follows:

$$iQDeep\,score_{high-resolution} = \frac{N1 + N2 + N4 + N8}{4} \tag{3.7}$$

where $N1$, $N2$, $N4$, and $N8$ represent the fraction of residues estimated to be within 1, 2, 4, and 8Å from the corresponding residues in the experimental structure. The estimated $iQDeep\,score_{standard-resolution}$score is equivalent to the GDT-TS score, ranging between 0 and 1 with a higher score indicating better model accuracy.

3.6.2 Protein structure prediction accuracy on CASP15 datasets

Supplementary Figure S1 presents the distribution of prediction accuracy of two recent protein structure prediction methods AlphaFold2 [2] and RoseTTAFold [9], on a common set of 60 targets. AlphaFold2 demonstrates high accuracy with an average GDT-TS score of 0.812, ranging from a maximum of 0.992 to a minimum of 0.378. Although not as accurate as AlphaFold2, RoseTTAFold approaches the accuracy of AlphaFold2 with an average GDT-TS score of 0.653, with a maximum score of 0.960 and a minimum score of 0.257.

However, when GDT-HA is used as the ground truth, the prediction accuracy is not as high as that of GDT-TS. AlphaFold2 has an average GDT-HA score that is almost 18% lower than its GDT-TS score, while RoseTTAFold's average GDT-HA score is around 27% lower than its GDT-TS score. This is due to the fact that GDTHA is more sensitive to small structural errors and penalizes larger deviations from the experimental structures [98, 107, 108]. Therefore, focusing on the GDT-HA metric for the predicted structure should further

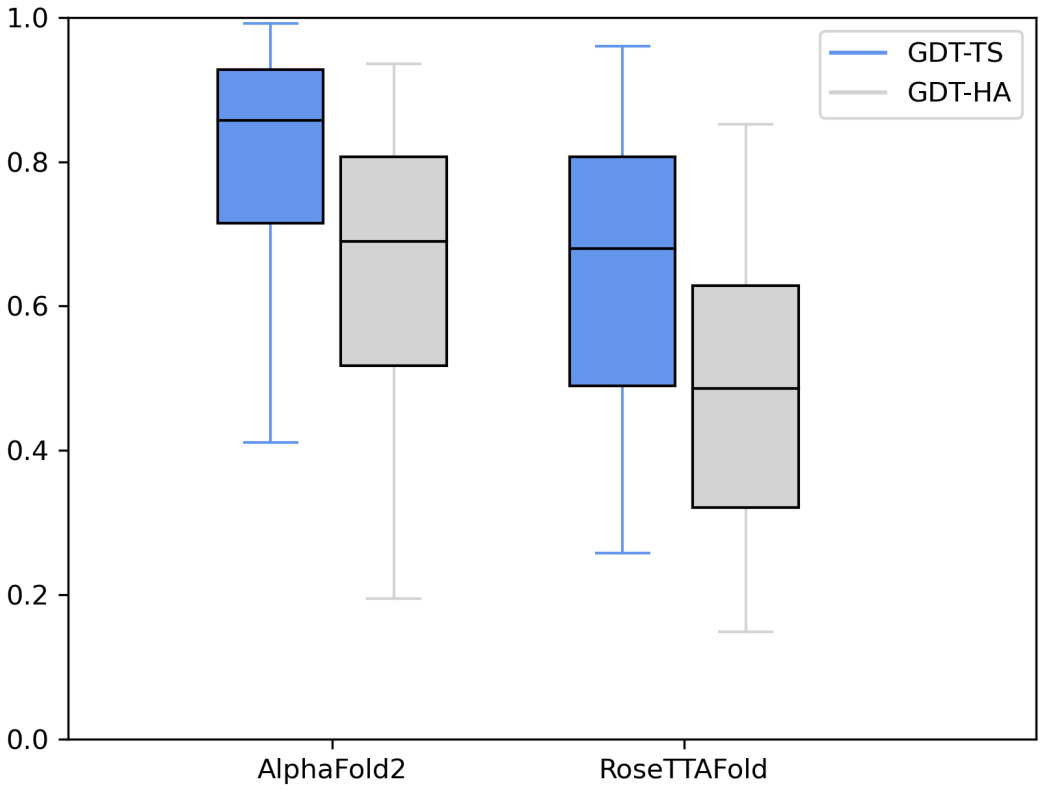

Figure 3.4: Protein structure prediction accuracy of AlphaFold2 and RoseTTAFold on CASP14 targets.

help in better evaluating the quality of highly accurate predicted structures, and thus helps in further improving prediction accuracy. The results highlight the need for high-resolution protein scoring methods that can accurately estimate the GDT-HA score.

Chapter 4

DeepRefiner: high-accuracy protein structure refinement by deep network calibration

4.1 Abstract

The DeepRefiner webserver, freely available at http://watson.cse.eng.auburn.edu/DeepRefiner/, is an interactive and fully configurable online system for high-accuracy protein structure refinement. Fuelled by deep learning, DeepRefiner offers the ability to leverage cutting-edge deep neural network architectures which can be calibrated for on-demand selection of adventurous or conservative refinement modes targeted at degree and consistency of refinement. The method has been extensively tested in the Critical Assessment of Techniques for Protein Structure Prediction (CASP) experiments under the group name "Bhattacharya-Server" and was officially ranked as the No. 2 refinement sever in CASP13 (second only to "Seok-server" and outperforming all other refinement severs) and No. 2 refinement sever in CASP14 (second only to "FEIG-S" and outperforming all other refinement severs including "Seok-server"). The DeepRefiner web interface offers a number of convenient features, including (i) fully customizable refinement job submission and validation; (ii) automated job status update, tracking, and notifications; (ii) interactive and interpretable web-based results retrieval with visual and quantitative analysis; and (iv) extensive help information on job submission

and results interpretation via web-based tutorial and help tooltips.

4.2 Introduction

Deep learning has transformed protein structure prediction. Recent editions of the Critical Assessment of Techniques for Protein Structure Prediction (CASP) experiments have witnessed a major breakthrough in accurately predicting the structure of a protein from sequence information through the application of advanced deep neural network architectures [63, 64, 109]. However, a predicted structure can still deviate from the experimental structures in terms of the accuracy of the backbone positioning or the side-chain conformation or both [110]. The goal of protein structure refinement is to increase the accuracy of such a moderately accurate starting structure by driving it towards the experimental quality. Some of the most successful approaches for structure refinement rely on large-scale conformational search for low energy structures [7, 111, 112], which are time-consuming and computationally expensive. To make structure refinement both accurate and fast, we proposed refineD [96] which employed deep learning models to estimate residue-level errors from a starting structure and then minimized the cumulative error through inexpensive energy-minimization-based restrained relaxation for improved structure refinement. Due to the advantages associated with computationally efficient energy minimization guided by deep learning over time-consuming conformational search, several recent studies have sought to guide refinement using deep learning [113, 114]; even though the full-fledged versions of these methods are not yet publicly available. As such, a robust and publicly accessible webserver that can perform high-accuracy structure refinement in a computationally efficient manner guided by deep learning has the potential for broad dissemination and a field-wide impact. With rapid new developments in the field, however, the residue-level error estimators used in our original refineD method no longer represents the state of the art. In particular, the recent CASP experiments [110] have witnessed significant new progress in inter-residue

distance prediction through cutting-edge deep neural network training in conjunction with metagenomic sequencing. Thus, integrating distance information for improved residue-level error estimation combined with the power of state-of-the-art deep learning models is critically important to further improve protein structure refinement. Moreover, the ability to leverage deep network architectures that can be calibrated for on-demand selection of adventurous or conservative refinement modes targeted at degree and consistency of refinement, can enhance the versatility of such a method to a wide range of use cases.

Here we present DeepRefiner, an interactive and fully configurable webserver for high-accuracy protein structure refinement by deep network calibration. DeepRefiner first estimates residue-level errors from a starting structure using an ensemble of advanced deep neural network architectures and subsequently minimizes the cumulative error through energy-minimization-based restrained relaxation, leading to five refined structures. The advanced error estimation module in DeepRefiner employs a high-resolution version of our successful application of very deep and fully convolutional residual neural networks [36] for distance-based protein model quality estimation [21] at finer-grained error thresholds trained specifically for structure refinement. DeepRefiner offers an interactive user interface that takes a starting structure in PDB format as input and outputs five refined structures along with their global and local quality estimations, comparison to the starting structure, and breakdown of residue-wise structural features. The customizable DeepRefiner interface provides (i) choice of cutting-edge deep neural network architectures for estimating residue-level errors including deep conditional neural fields and deep residual neural networks; (ii) on-demand selection of adventurous or conservative refinement mode by calibrating the ensemble of deep networks; (iii) comprehensive post-refinement analysis using MolProbity [115], GOAP [116], OPUS-PSP [117], DFIRE [118], and RWplus [119]; (iv) fully automated job status update, tracking, and notifications; (v) interactive and interpretable web-based results; and (vi) extensive help information on job submission and results interpretation via web-based tutorial and help tooltips. Our method was rigorously tested in the most recent CASP refinement

experiments [120] under the group name "Bhattacharya-Server" and was officially ranked as the No. 2 refinement sever in CASP13 (second only to "Seok-server" and outperforming all other refinement severs) and No. 2 refinement sever in CASP14 (second only to "FEIG-S" and outperforming all other refinement severs including "Seok-server"). DeepRefiner web-server is freely available at http://watson.cse.eng.auburn.edu/DeepRefiner/.

4.3 MATERIALS AND METHOD

4.3.1 Overview of the DeepRefiner pipeline

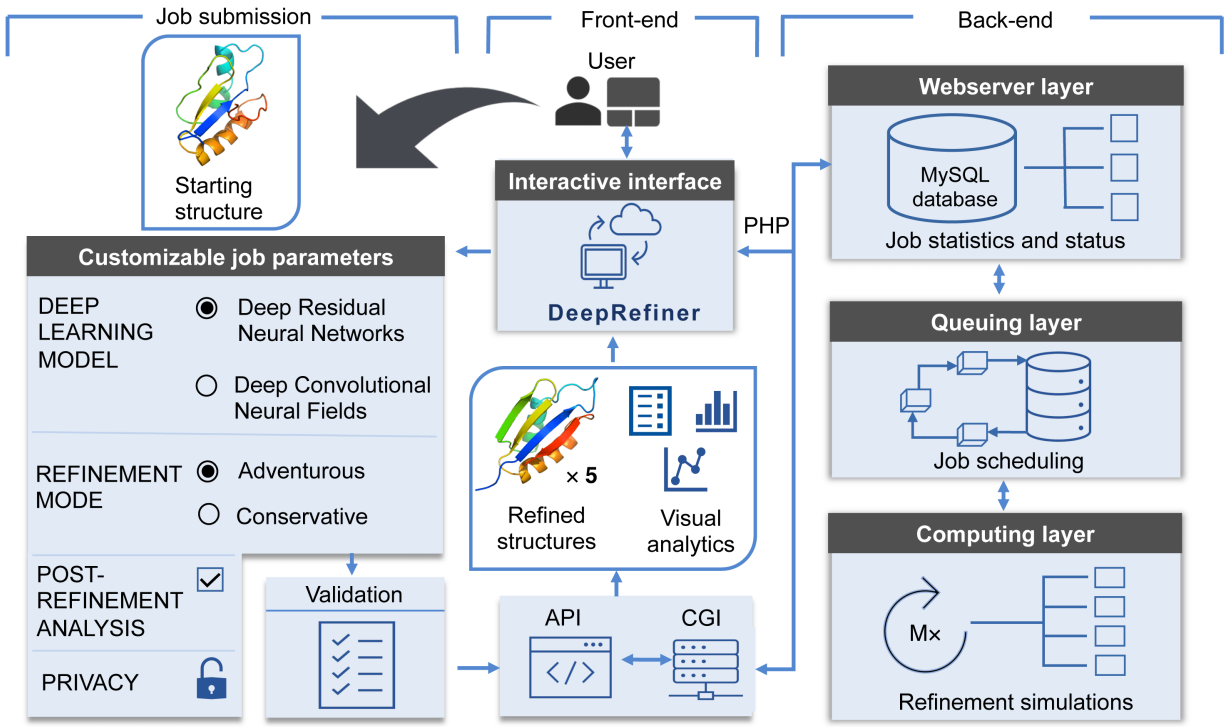

Figure 4.1: The flowchart of the DeepRefiner pipeline consisting of the webserver front-end module for submitting customizable refinement jobs through the interactive web interface and retrieving the results with interactive visual analysis, and the back-end module that processes the refinement jobs.

Figure 4.1 shows the flowchart of the DeepRefiner pipeline consisting of the webserver front- and back-end modules. The front-end module offers an interactive web-based interface that

lets the user submit customizable refinement jobs, readily processes and validates user inputs, dynamically shows the job status and progress, presents statistics of the processed job, and provides interactive visual analytics of the results; while the back-end module runs the refinement jobs. Users can choose to protect the privacy of their jobs by submitting a private refinement job where the refinement results will only be accessible to the submitter.

Front-end module. The front-end module of the webserver provides a web-based job submission interface where the user needs to provide only a starting structure in PDB format and a job name as mandatory inputs. The interface offers a wide range of options for customizing a refinement job including the ability to select deep network architectures, calibrate the refinement modes for on-demand selection of adventurous or conservative refinement targeted at degree and consistency of refinement, perform comprehensive post-refinement analysis, and protect job privacy. Two deep neural network architectures for residue-level error estimation are available, both independently supporting on-demand selection of adventurous or conservative refinement modes by calibrating the model ensemble. The submitted refinement job is then dynamically validated for consistency and passed on to the back-end via Common Gateway Interface (CGI).

Back-end module. The back-end of the webserver consists of three sequentially interdependent layers including the webserver layer, queuing layer, and computing layer. The webserver layer directly interacts with the front-end and maintains job statistics and status using a MySQL database. The queuing layer performs job scheduling by implementing a first-in-first-out (FIFO) job queue. Additionally, it continuously interacts with the webserver layer to dynamically update job status and subsequently communicates with the front-end through CGI. Once the queuing layer releases a job for execution, the job starts running in the computing layer. The computing layer executes the job based on the supplied job parameters by first employing the selected deep neural network architecture for estimating the residue-level errors from the starting structure and then performing either adventurous or conservative

refinement through energy-minimization-based restrained relaxation by calibrating the chosen model ensemble to generate five refined structures. After completion of the refinement job, the webserver performs comprehensive post-refinement analysis on the refined structures by estimating their global and local qualities; evaluating and reporting the scores of several knowledge-based statistical potentials including GOAP, OPUS-PSP, DFIRE, and RWPlus [116, 117, 118, 119]; performing MolProbity [115] analysis for assessing the physical realism; and comparing to the staring structure in terms of backbone and side-chain positioning as well as the consistencies between structural properties such as secondary structure and solvent accessibility. The results are returned to the front-end module for interactive and interpretable web-based visual analytics (see Supplementary Figure 4.3) with an email notification sent to the user, if an email address is provided.

4.3.2 Deep network calibration

Architectures of the deep learning models. DeepRefiner offers the choice of two deep neural network architectures for estimating residue-level errors including deep conditional neural fields (DeepCNF)[121, 122] and deep residual neural networks (ResNet)(11). DeepCNF architecture was employed in our original refineD method [96] to classify every residue of the starting structure to be within four fine-grained error thresholds of 0.5, 1, 2, and 4Å by independently training an ensemble of four DeepCNF classifiers. Collectively, the four classifiers result in residue-level ensemble error classifications. The featurization for representing the residues in the starting structure includes sequence profile, consistency between predicted and observed structural properties (secondary structure and solvent accessibility), and biophysical energy terms. Our newly trained ResNet ensemble classifiers incorporate distance information as additional features to perform residue-level ensemble error classifications at the same fine-grained error thresholds of 0.5, 1, 2, and 4Å. The ResNet classifiers represent a high-resolution version of our successful application of distance-based protein model qual-

ity estimation [21], while specifically targeting finer-grained error thresholds for structure refinement (see the detailed description in section 4.11.1 in the Supporting Information).

Calibrating the model ensemble The residue level ensemble error estimates are then converted into multi-resolution probabilistic restraints weighted by their associated likelihood values and applied on the C atom of the starting structure in the form of Rosetta Coordinate Constraint with FLAT_HARMONIC function at 0.5, 1, 2, and 4Å thresholds in conjunction with the REF15 scoring function of Rosetta [123]. Subsequently, energy-minimization-based restrained relaxation is iteratively employed for structure refinement (see section 4.11.2). All restraints corresponding to the four thresholds can be simultaneously applied in a cumulative manner for conservative refinement mode aimed at achieving consistently positive refinement. Alternatively, restraints can be applied in a non-cumulative manner independent of each other for adventurous refinement mode aimed at producing higher degree of structural changes. In summary, calibration of the model ensemble controls the characteristics of the restrained relaxation, thus affecting the degree of conformational change that can be used for achieving on-demand conservative or adventurous structure refinement.

4.4 RESULTS

4.4.1 Blind performance assessment in CASP

The refinement protocol employed in DeepRefiner has been extensively tested in the refinement category of CASP13 and CASP14 in a strictly blind manner under the group name "Bhattacharya-Server" and was ranked highly among all refinement servers. Table 4.1 shows the performance comparison of "Bhattacharya-Server" with other participating server groups under the refinement category of CASP13 and CASP14 in terms of the sum of overall Z-scores calculated as the weighted sum of Z-scores for GDT-HA (22), GDC-sc [124], RMSD [125], SphereGrinder [91], and MolProbity [115], following the same methodology adopted in

Table 4.1: Performance comparisons of server groups participating in the refinement category of CASP13 and CASP14. Groups are sorted by descending sum of overall Z-scores.

	Group name	Group #	Sum overall Z-score	Rank sum overall Z-score
CASP13	Seok-server	156	21.686	1
	Bhattacharya-Server	102	13.125	2
	YASARA	004	12.976	3
	MUFold_server	312	10.895	4
	3DCNN	359	0.701	5
CASP14	FEIG-S	013	35.344	1
	Bhattacharya-Server	149	21.822	2
	Seok-server	070	18.404	3
	MULTICOM-CLUSTER	075	12.312	4
	MUFOLD	081	4.178	5

the previous rounds of CASP refinement assessments [126]. "Bhattacharya-Server" was officially ranked No. 2 among all other refinement server groups in both CASP13 (second only to "Seok-server" and outperforming all other refinement severs) and CASP14 (second only to "FEIG-S" and outperforming all other refinement severs including "Seok-server"). We report the per-target Z-score in the supplementary information (see Tables 4.2 – 4.3 for per-target Z-scores broken down by each accuracy metric involved in the calculation of the overall Z-score). We further analyze the degree of structural refinement attained by 'Bhattacharya-Server' for CASP13 and CASP14 refinement targets in terms of various accuracy measures including GDT-HA, GDC-sc, and MolProbity scores with respect to length and accuracy of the starting structures (in terms of GDT-HA) considering the best submission. The results demonstrate that most promising refinement cases are generally observed for smaller targets having length less than 100 residues and those in the medium range of starting accuracies having starting GDT-HA scores between 40 and 60 units (see Supplementary Figures 4.4 and 4.5). The DeepCNF-based error estimation module used in 'Bhattacharya-Server' for both CASP13 and CASP14 shall further improve due to the incorporation of advanced

ResNet-based ensemble error classifiers, ultimately improving the refinement performance of DeepRefiner.

4.4.2 Case Study

In Figure 4.2, we present the refinement results for four representative CASP targets including two targets R0957s2 and R1009 from CASP13; and two targets R1085-D1 and R1065s2 from CASP14. DeepRefiner alternates between adventurous or conservative refinement modes by deep network calibration using either ResNet- or DeepCNF-based error estimation. We also submit these four targets to the popular refinement webservers including GalaxyRefine [127], GalaxyRefine2 [128], 3Drefine [107], and ModRefiner [129] for performance comparison. In all cases, DeepRefiner outperforms the other servers by consistently producing positive refinement. The adventurous refinement modes lead to noticeable structural improvements with higher degree of refinement compared to the other methods, whereas the conservative modes yield modest but positive refinement with higher consistency even when all other methods produce negative refinement.

4.5 WEB SERVER

4.5.1 Hardware and software

The server runs on a Linux cluster of 2.20-GHz Intel Xeon E5-2698 v4 20-core processors. The web application uses the PHP scripting language, JavaScript programming language, and MySQL database. WebGL-based molecular visualization package 3Dmol.js [100] is used to visualize the protein structures. The DeepRefiner pipeline is implemented using Python. The webserver is compatible with most modern web browsers including Mozilla Firefox, Google Chrome, Safari, and Microsoft Edge

4.5.2 Input and output

The mandatory inputs are a job name and a starting structure for refinement in PDB format. The customizable DeepRefiner interface provides users the ability to fully configure various optional job parameters including deep learning model, refinement mode, post-refinement analysis, and job privacy. An optional email address can also be provided for automated status update of the refinement job via email. The number of residues in the starting structure is limited to 500 for computational efficiency. The average run time is several hours after the job enters the running state. Five refined structures ranked based on the estimated global qualities, MolProbity scores, and various statistical potentials are visualized in the interactive web interface and are downloadable in the PDB format. Information on structural comparison between the starting structure and the refined structures is provided in terms of GDC-sc, GDT-HA, GDT-TS, and C -RMSD. Structural agreement between the starting structure and the refined structures in terms of secondary structure and solvent accessibility is shown in a visually interpretable manner. Estimated local quality containing the residue-level error estimates are visualized in a graphical format for the identification of potentially unreliable local regions. The full set of results, including the refined structure and text files containing the refinement analysis, can be downloaded as a compressed zipped archive.

4.6 CONCLUSION

DeepRefiner presents a publicly available webserver for accurate and efficient protein structure refinement. DeepRefiner leverages cutting-edge deep neural network architectures that can be calibrated for on-demand selection of adventurous or conservative refinement modes targeted at degree and consistency of refinement. The method was successful in blind refinement experiments in CASP13 and CASP14. DeepRefiner offers an interactive and versatile

web interface for the submission, monitoring, results retrieval, and analysis of refinement jobs in order to drive a moderately accurate starting structure towards the experimental quality.

4.7 AVAILABILITY

DeepRefiner webserver is freely available at http://watson.cse.eng.auburn.edu/DeepRefiner/.

4.8 Supplementary data

Supplementary Data are available at NAR online.

4.9 Acknowledgement

This work was made possible in part by a grant of high performance computing resources and technical support from the Alabama Supercomputer Authority and the Extreme Science and Engineering Discovery Environment (XSEDE).

4.10 Funding

National Institute of General Medical Sciences [R35GM138146]; National Science Foundation [IIS2030722, DBI1942692]. Funding for open access charge: National Institute of General Medical Sciences.

Conflict of interest statement. None declared.

4.11 Supplementary Information

4.11.1 Feature generation for the advanced error estimation module of DeepRefiner.

The advanced error estimation module of DeepRefiner based on deep residual neural networks (ResNet) uses both sequence-based predicted and structural features as well as the features based on the consistency between predicted and observed structural properties to represent each of the residues in the structure. The overall feature generation strategy is briefly described below:

Generation of Multiple Sequence Alignment (MSA): To generate Multiple Sequence Alignment (MSA), we perform three iterations of HHblits [68] search with an E-value inclusion of $10^{-}3$ against the protein sequence database Uniclust30 [69] with a query sequence coverage of 10% and pairwise sequence identity of 90%. Subsequently, the generated MSA is used for predicting various sequence-based features including secondary structure, solvent accessibility, and inter-residue distance maps.

Generation of Position-Specific Scoring Matrix (PSSM): We generate PSSM from the query sequence by performing a search against the NR database using PSI-BLAST v2.2.26 software [74] with an E-value of 0.001.

Prediction of secondary structure, solvent accessible surface area, and backbone dihedral angles: We use SPIDER3 [103] to predict three-class secondary structures (H/E/C), solvent accessible surface area, and backbone dihedral angles (ϕ, ψ) from the query sequence. We extract the observed secondary structure and solvent accessible surface area using DSSP [104] or STRIDE [130].

Prediction of inter-residue distance maps: We use DMPfold [52] to predict the inter-residue

distance maps. We obtain the rawdistpred.current file generated by DMPfold as the initial prediction without any iterative refinement. For each of the interacting residue pairs, the distance bins and their associated likelihoods are subsequently used for generating distance-based features.

Featurization: We use 23 features for each residue in the structures that include:
(i) Sequence profile conservation score: We extract information per-position score for each residue in the starting structure from the generated PSSM and subsequently apply the sigmoid function to normalize the scores between 0 and 1 to be used as a feature. *(ii) The number of effective sequences:* We use calNf_ly program [65] that accepts Multiple Sequence Alignment (MSA) to calculate the number of effective sequences (NEFF) by summing up the nth reciprocal of the total number of sequences in the MSA with sequence identity greater than 80% to the corresponding sequence. The resulting NEFF is used as a feature. *(iii) Agreement between predicted and true secondary structure and solvent accessibility:* We calculate the residue-specific sequence-structure agreement between the predicted and observed secondary structure and solvent accessibility. For each of the residues in the structure, we assign a feature value of 1 if both predicted and observed secondary structures agree and 0 otherwise. Furthermore, we calculate the squared error between the predicted and observed solvent accessibility and subsequently apply the sigmoidal transformation to normalize the squared error. We then use them as two features. *(iv) Angular RMSD:* We calculate backbone dihedral angles (ϕ, ψ) for each of the residues in the structure. Afterwards, we calculate the angular root mean square deviation between the observed dihedral angles (ϕ_o, ψ_o) and SPIDER3 (4) predicted backbone dihedral angles (ϕ_o, ψ_o) as follows:

$$AngularRMSD = \sqrt{\frac{1}{n}\sum_{i}(min(x_{2i} - x_{i1}, 2\pi - |x_2 i - x_i 1|))^2} \tag{4.1}$$

$$NormalizedAngularRMSD = \frac{1}{1 + \left(\frac{AngularRMSD}{\pi/4}\right)^2} \tag{4.2}$$

Where x_o and x_p represent the vector of observed and predicted ϕ or ψ dihedral angles for n residues in the structure. We subsequently normalize the angular RMSD values and use them as two features.

(v) Distance-based weighted histogram alignment score: We use DMPfold predicted distance histogram (i.e., rawdistpred.current file) to extract inter-residue distance distribution at 5 evenly distributed distance intervals of 6, 8, 10, 12, and 14Å by summing up the likelihoods values of all the bins below a specific distance threshold. To minimize noise, we discard all residue pairs with likelihood values below 0.2. We also calculate the observed distance histogram at 6, 8, 10, 12, and 14Å from the structure to perform eigen-decomposition and dynamic programming-based alignment between the predicted and observed histogram resulting in 5 distance alignment scores. We subsequently multiply the raw alignment scores at 6, 8, 10, 12, and 14Å with empirically selected weights of 0.10, 0.25, 0.30, 0.25, and 0.10, respectively, and use them as 5 distance-based features.

(vi) Rosetta centroid energy terms: We calculate 12 Rosetta centroid energy terms [76, 77] including the residue environment (env), residue pair interactions (pair), c density (cbeta), steric repulsion (vdw), radius of gyration (rg), packing (cenpack) density, contact order (co), statistical potentials for secondary structure formation (hs_pair, ss_pair, sheet, rsigma), and centroid hydrogen bonding (cen_hb). Once again, we use the sigmoid function to normalize the scores between 0 and 1 before using as features.

4.11.2 Training an ensemble of deep residual neural networks at finer-grained error thresholds.

The advanced version of residue-level error estimator used in DeepRefiner is an improvement over the deep networks utilized in our original refinement protocol refineD [96]. The advanced error estimation module of DeepRefiner employs an ensemble of deep residual neu-

ral networks (ResNet) trained at finergrained distance thresholds following a similar strategy as leveraged by our successful application of protein model quality estimation method QDeep [21]. Specifically, each ResNet classifier consists of 13 residual blocks, each consisting of three 1D convolutional layers with a kernel size of 1×1, 1×3, and 1×1, which is similar to the bottleneck design [36]. Each 1D convolutional neural network layer uses an L2- norm regularization with a weight decay parameter of 0.0001 to prevent overfitting (14). 1×1 layers, also known as "bottleneck layers" at the end of each residual block are used to preserve the dimension of the network by reducing and restoring the dimensionality of the input feature vector. Each classifier takes an input with a shape of $L \times 483 \times 1$ to its first convolutional layer with a kernel and filter size of 1×7 and 64×1, which is subsequently transformed into a 64-dimensional feature vector using 1×1 convolutional layer of the first residual block with a filter size of 64×1. Each residual block in the network also establishes a "shortcut or skip connection" between the input and the output layer, performing as an identity mapping between the input and the output layer. We adopt a multi-stage design of the ResNet model by sequential stacking 3 consecutive stages having 3, 4, and 6 residual blocks in the first, second, and third stage, respectively, with an output channel dimension of 128, 256, and 512, respectively. At the end of the residual blocks, we use an average pooling layer with a pool size of 2 that helps in faster computation by halving the number of parameters. Afterwards, we add a flatten layer that accepts the pooled features and transformed them into a 1D vector. The fully connected (dense layer) layer at the end accepts the 1D vector and applies sigmoidal activation to return a probability value for residue-level error classification.

To perform residue-level ensemble error classification, we independently train four binary classifiers that can estimate the residue-level C errors at four error thresholds of 0.5, 1, 2, and 4Å, analogous to GDT-HA. The ensemble classifiers leverage the same features for representing each residue in the structures as described before. We assign a binary label to each residue of the starting structure by calculating the Euclidian distance error between

the C atom of a residue and the corresponding aligned residue in the experimental structure after performing the sequence-dependent analysis between the model and the corresponding experimental structure using the LGA program [67]. While assigning labels, we consider four fine-grained error thresholds of 0.5, 1, 2, and 4Å and assign a label of 1 if the distance error is within a threshold and 0 otherwise. We train the ensemble classifiers by using a maximum number of epochs of 120 and an optimal batch size of 64 that best fits the GPU limit. To avoid overfitting, we use EarlyStopping callback of Keras [82] with a patience value of 20. We optimize the model using the binary crossentropy loss function and first-order gradient-based Adam optimizer. After training, each binary classifier estimates the residue-level errors at four fine-grained thresholds of 0.5, 1, 2, and 4Å. Collectively, the set of four classifiers results in residue-level ensemble error classifications.

4.11.3 Iterative restrained relaxation for guiding protein structure refinement.

After performing residue-level ensemble error classifications, we subsequently convert the error estimates into multi-resolution probabilistic restraints weighted by their associated likelihood values and apply on the C atom of the starting structure in the form of Rosetta Coordinate Constraint with FLAT_HARMONIC function at 0.5, 1, 2, and 4Å thresholds in conjunction with the Rosetta's full atom Energy Function [123] toperform restrained relaxation. We use Rosetta's FastRelax application [131, 132], which inherently performs several rounds of side-chain repacking and gradient-based backbone minimization. We generate a total of 100 refined models by iteratively applying restrained relaxation. Restraints are imposed simultaneously in a cumulative manner for conservative refinement mode aimed at achieving consistently positive refinement, whereas imposing restraints in a non-cumulative manner independent of each other leads to adventurous refinement mode aimed at producing higher degree of structural changes.

4.12 Global and local quality estimation of the refined structures.

DeepRefiner uses the residue-level ensemble error classifications to estimate the global and local quality of the refined structures. Ensemble of DeeCNF and ResNet classifiers estimate the likelihood of each residue in the structure to be within rÅ (r ϵ {0.5, 1, 2, 4} Å) threshold. A likelihood cutoff of 0.5 is then used for classifying a residue as 1 and 0 otherwise. Subsequently, the global quality score of a refined structure is calculated analogous to GDT-HA by a weighted combination of the residue-level ensemble error classifications at 0.5, 1, 2, and 4Å thresholds as follows:

$$global\_quality = \frac{N_{0.5} + N_1 + N_2 + N_4}{4 \times L} \tag{4.3}$$

Where L is the length of the structure and $N_{0.5}$, N_1, N_2, and N_4 are the number of residues predicted to be aligned with their corresponding residues in the native structure at 0.5, 1, 2, and 4Å thresholds. Thus, global_quality score ranges between [0, 1], with a higher score indicating better global quality.

DeepRefiner further uses the residue-level error estimates at 0.5, 1, 2, and 4Å thresholds to estimate the residue-level quality score by employing an inverse S-score calculation as follows:

$$local\_quality(r) = \frac{\sum_{r\epsilon\{0.5,1,2,4\}} \left(\frac{r \times (\sqrt{1-p_r})}{p_r} \right)}{4} \tag{4.4}$$

Where pr denotes the probability of i_{th} residue in the structure, predicted by a specific classifier at rÅ (r ϵ 0.5, 1, 2, 4 Å) threshold.

4.13 Refinement evaluation.

To evaluate the performance of "Bhattacharya-Server" participating as a server group in CASP13 and CASP14 refinement category, we use a combination of several accuracy measures including GDT-HA [97], RMSD [125], GDC-sc [124], SphereGrinder [91], and MolProbity [115]. Specifically, we calculate the Zscores of various accuracy measures as routinely performed by CASP assessors [133] by following a two step procedures considering all top-ranked submissions. In the first step, all submissions with a Z-score lower than -2 are discarded. In the second step, we use a penalty threshold of 0 in calculating the final Zscore using the scores obtained from the first step. The penalty threshold of 0 assigns a score of 0 to any model having a Z-score less than 0. We subsequently rank the participating server groups based on the sum of overall Z-score. Following previous CASP refinement assessment [126], the overall Z-score is calculated as the weighted sum of Z-scores of GDT-HA, RMSD, GDC-sc, SphereGrinder, and MolProbity scores as follows:

$$Overall\, Z-score = \frac{4 \times Z_{GDT_{HA}} - Z_{rmsd} + Z_{GDC-sc} + Z_{sph_{GR}} - Z_{MP}}{8} \tag{4.5}$$

The above scoring scheme emphasizes the accuracy of backbone positioning as measured by GDT-HA score, which is widely used by the CASP assessors for refinement evaluation, while integrating other complementary aspects of refinement including side-chain quality, stereochemistry, and physical realism. The per-target Z-scores in terms of GDT-HA, RMSD, GDC-sc, SphereGrinder, and MolProbity scores as well as the overall Z-score calculated as the weighted sum of Z-scores for top server groups participating in CASP13 and CASP14 refinement experiments are reported in Table 4.2 and Table 4.3, respectively.

Table 4.2: Per-target Z-scores for Seok-server (156), Bhattacharya-Server (102), YASARA (004), MUFold_server (312), and 3DCNN (359) in CASP13 in terms of GDT-HA, RMSD, GDC-sc, SphereGrinder and MolProbity scores as well as per-target overall Z-score calculated as the weighted sum of Z-scores for GDT-HA, GDC-sc, RMSD, SphereGrinder, and MolProbity.

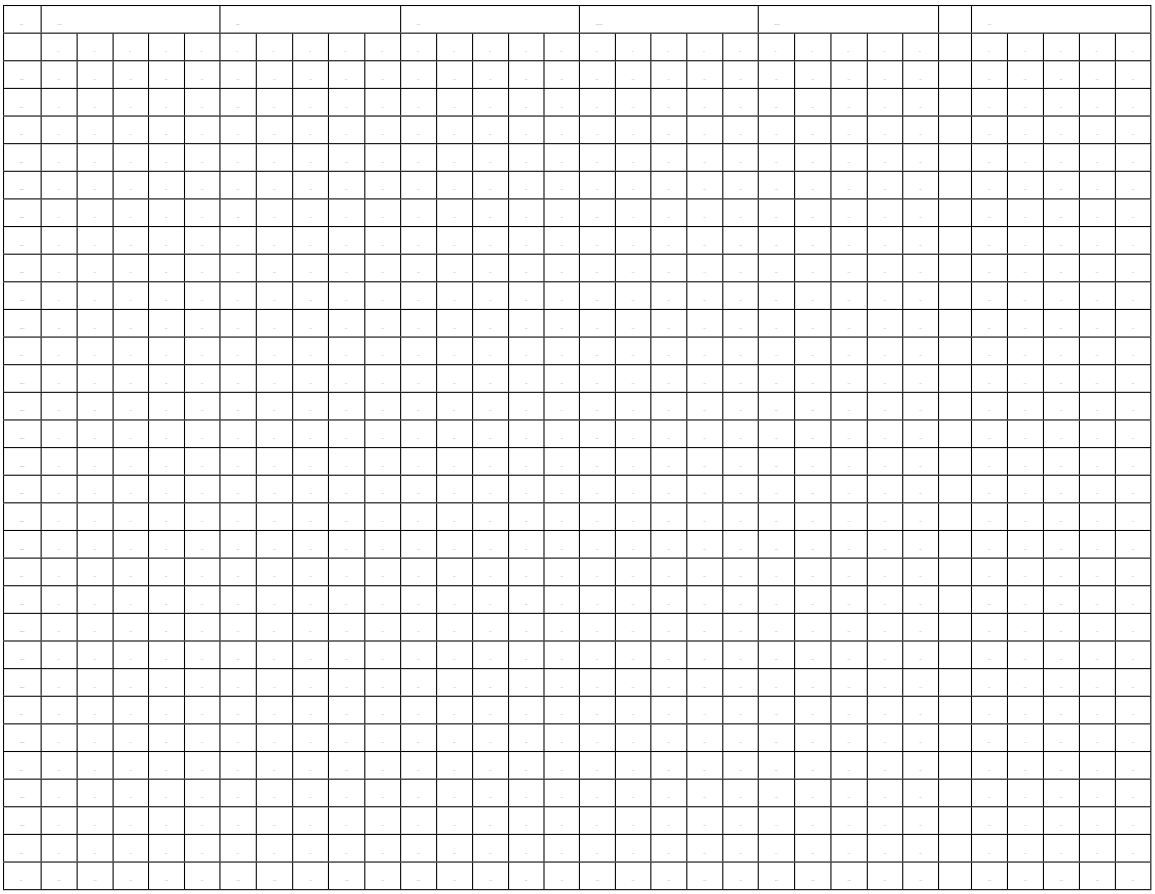

Table 4.3: Per-target Z-scores for FEIG-S (013), Bhattacharya-Server (149), Seok-server (070), MULTICOM-CLUSTER (075), and MUFOLD (081) in CASP14 in terms of GDT-HA, RMSD, GDC-sc, SphereGrinder and MolProbity scores as well as per-target overall Z-score calculated as the weighted sum of Z-scores for GDT-HA, GDC-sc, RMSD, SphereGrinder, and MolProbity.

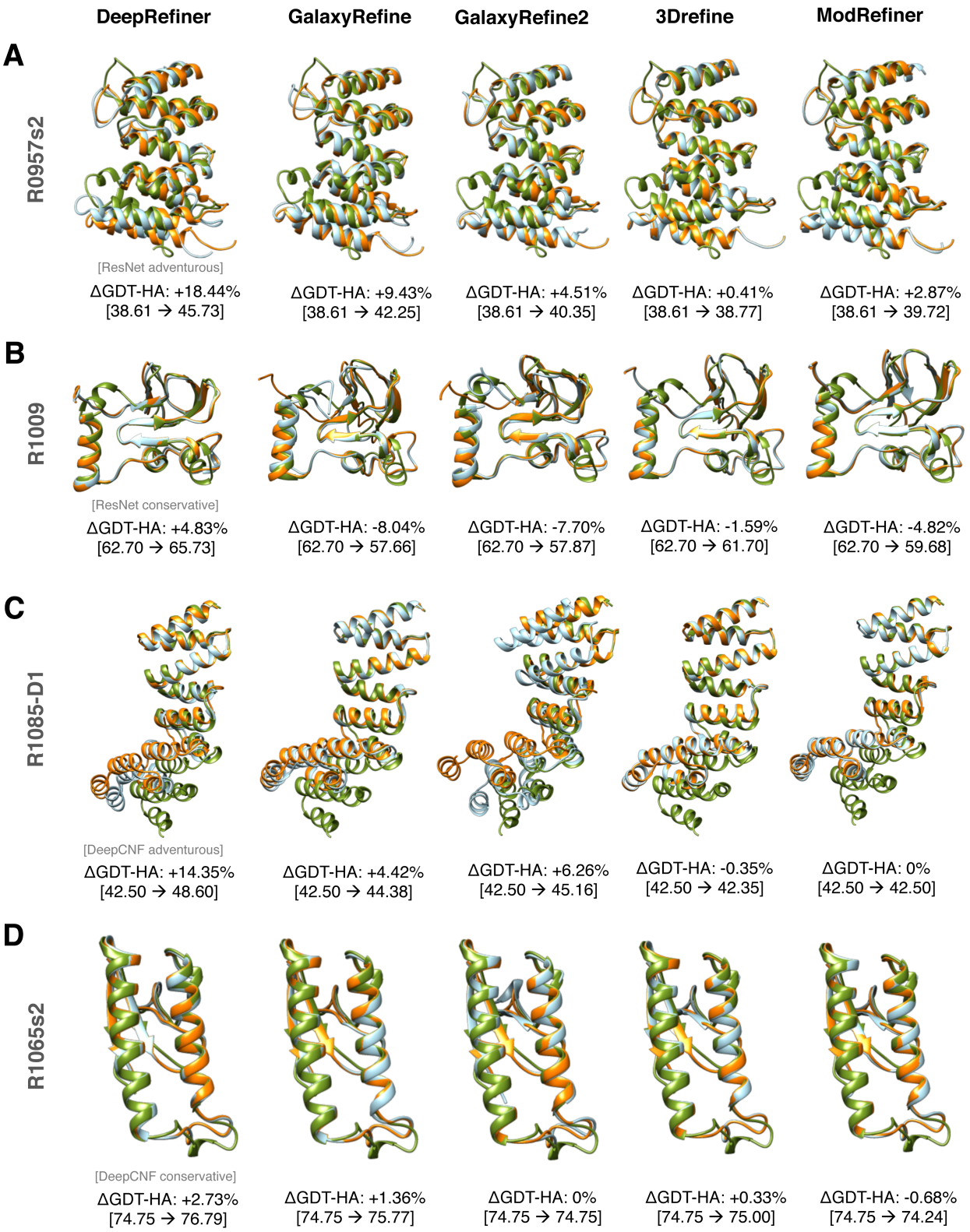

Figure 4.2: Representative refinement examples from four CASP refinement targets. DeepRefiner yields better refinement than other methods by deep network calibration using either ResNet- (A) R0975s2 and (B) R1009; or DeepCNF-based error estimation (C) R1085-D1 and (D) R1065s2.

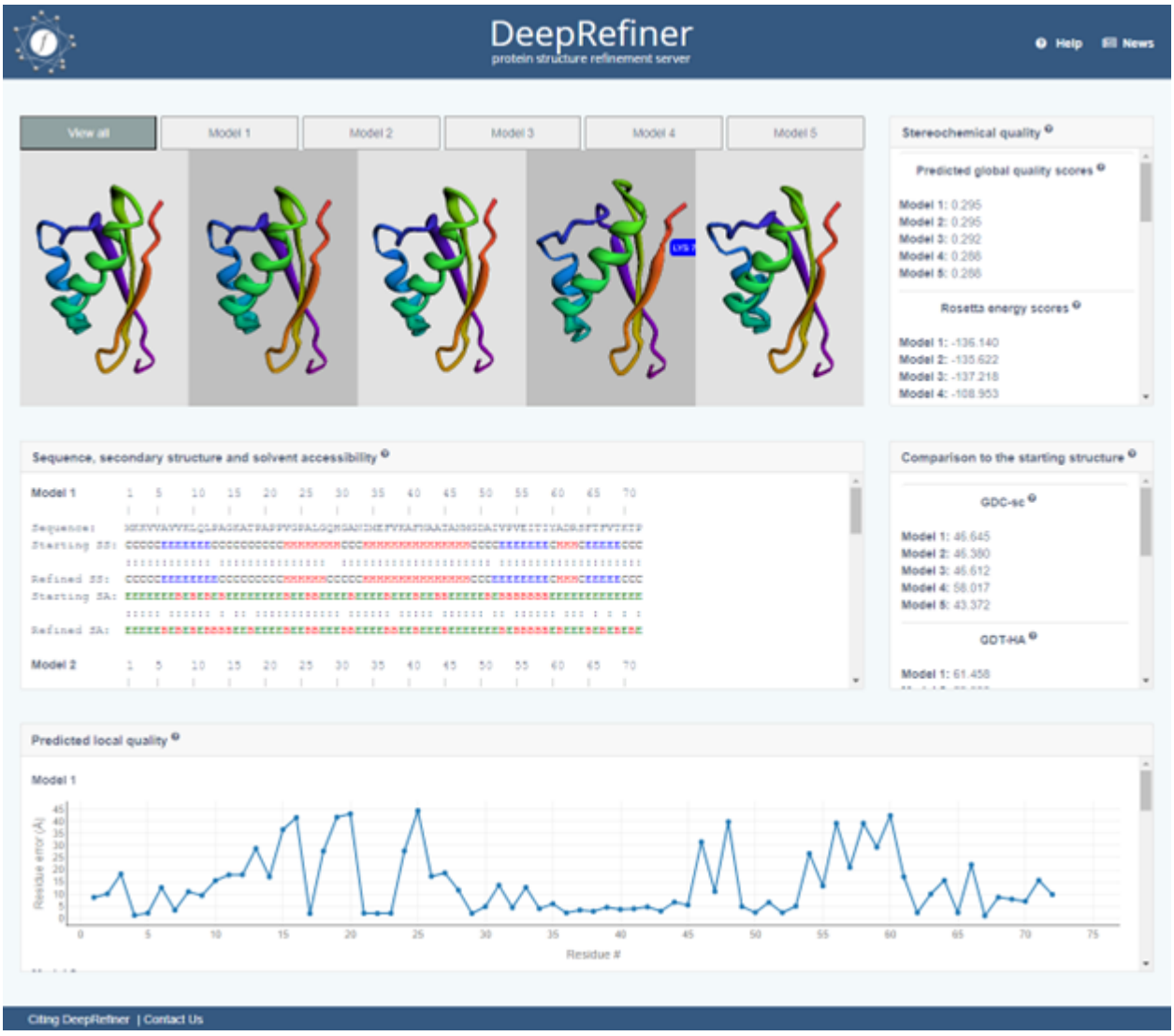

Figure 4.3: Interactive quantitative and visual analysis of the refinement results by DeepRefiner.

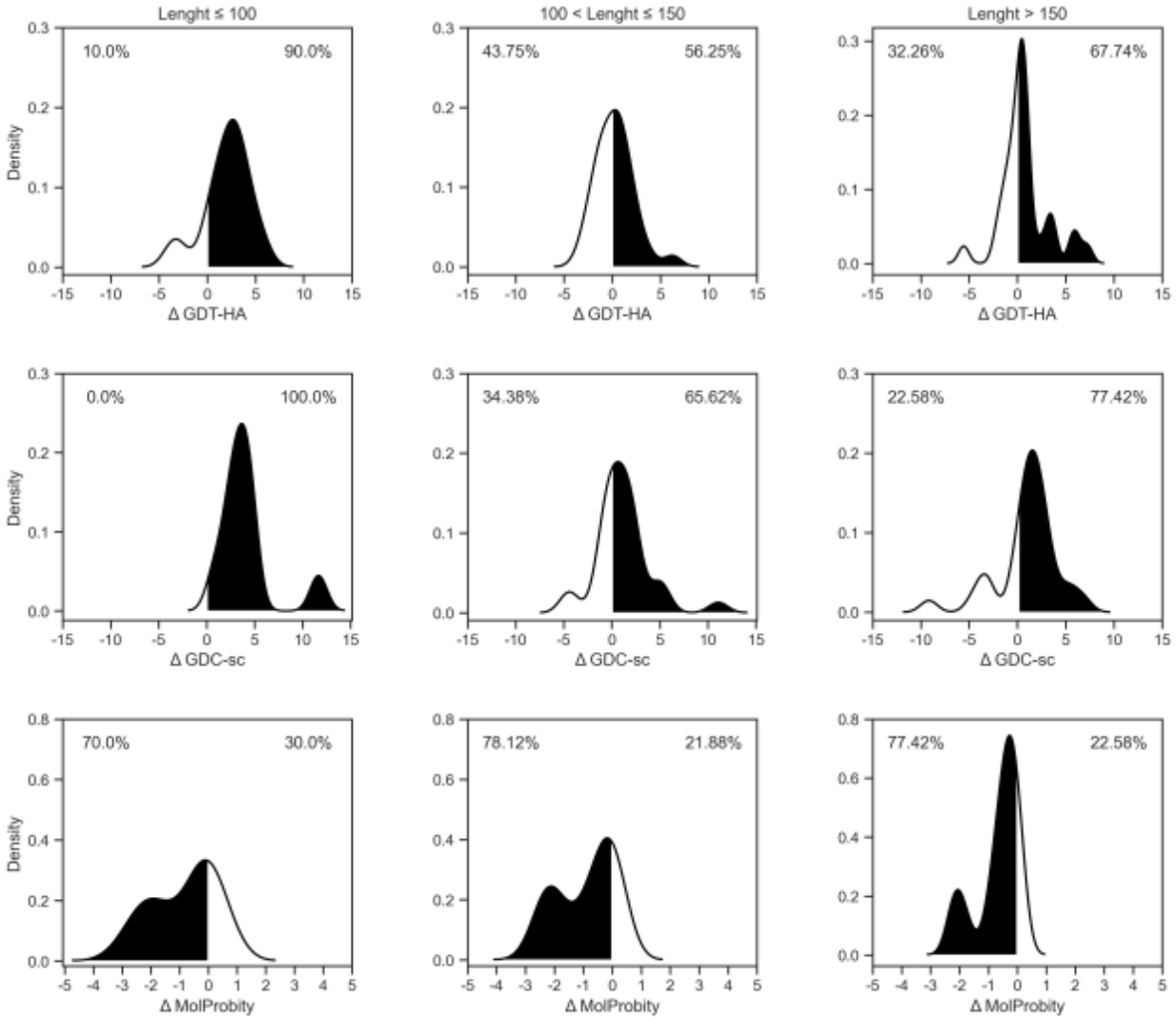

Figure 4.4: Degree of refinement for "Bhattacharya-Server" considering all CASP13 and CASP14 refinement targets grouped by various length bins. Distributions of ΔGDT-HA, ΔGDC-sc, and ΔMolProbilty scores for various length bins considering the best submission presented. Regions shaded in black illustrate improvement over the starting structure. Numbers above the distribution plots indicate the percentages of refinement successes and failures.

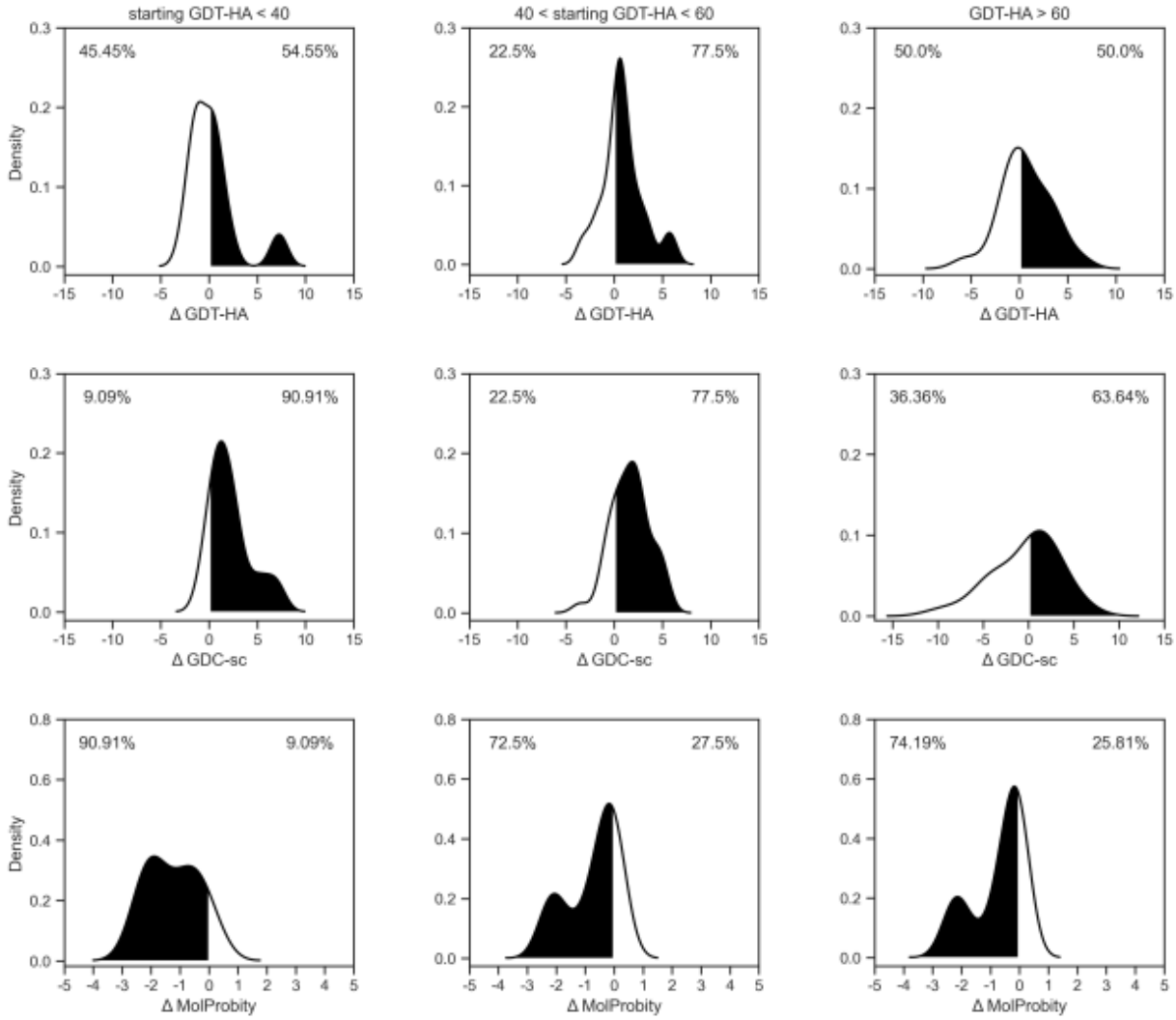

Figure 4.5: Degree of refinement for "Bhattacharya-Server" considering all CASP13 and CASP14 refinement targets grouped by various starting GDT-HA bins. Distributions of ΔGDT-HA, ΔGDC-sc, and ΔMolProbilty scores for various starting GDT-HA bins considering the best submission presented. Regions shaded in black illustrate improvement over the starting structure. Numbers above the distribution plots indicate the percentages of refinement successes and failures.

Chapter 5

PIQLE: protein-protein interface quality estimation by deep graph learning of multimeric interaction geometries

5.1 Abstract

Accurate modeling of protein–protein interaction interface is essential for high-quality protein complex structure prediction. Existing approaches for estimating the quality of a predicted protein complex structural model utilize only the physicochemical properties or energetic contributions of the interacting atoms, ignoring evolutionarily information or interatomic multimeric geometries, including interaction distance and orientations.

Here, we present PIQLE, a deep graph learning method for protein–protein interface quality estimation. PIQLE leverages multimeric interaction geometries and evolutionarily information along with sequence- and structure-derived features to estimate the quality of individual interactions between the interfacial residues using a multi-head graph attention network and then probabilistically combines the estimated quality for scoring the overall interface. Experimental results show that PIQLE consistently outperforms existing state-of-the-art methods including DProQA, TRScore, GNN-DOVE and DOVE on multiple independent test datasets

across a wide range of evaluation metrics. Our ablation study and comparison with the self-assessment module of AlphaFold-Multimer repurposed for protein complex scoring reveal that the performance gains are connected to the effectiveness of the multi-head graph attention network in leveraging multimeric interaction geometries and evolutionary information along with other sequence- and structure-derived features adopted in PIQLE.

An open-source software implementation of PIQLE is freely available at https://github.com/BhattacharyaLab/PIQLE.

5.2 Introduction

Protein-protein interactions are the actuators of numerous biological processes [134]. Despite the remarkable progress in predicting single-chain protein structures with very high degree of accuracy [2, 9, 135], modeling the structures of protein complexes remains challenging [136, 137, 138]. Traditional protein-protein docking approaches as well as recent deep learning-based protein complex structure prediction methods typically generate a number of candidate structural models and rank them based on estimated confidence scores to select the top-ranked model [137, 139, 140, 141, 142]. With the state-of-the-art protein structure prediction methods approaching near experimental accuracy on single-chain predictions, accurately modeling the protein-protein interaction interfaces is the key to successfully predicting the structures of protein complexes. As such, high-fidelity estimation of the modeling quality of protein-protein interaction interface from a computationally predicted complex structure is critically important for characterizing protein-protein interactions [143, 144].

Encouraging progress has been made in protein complex scoring and quality estimation. Physics-based approaches such as ZRANK [145] demonstrate effective scoring performance using the weighted sum of several energy terms including van der Waals force, hydrogen bonding, electrostatics, pair potentials, and solvation. ZRANK2 [146] further improves the scoring performance by optimizing certain energy terms used in ZRANK. In addition to physicsbased

approaches, state-of-the-art methods apply machine learning for the quality estimation of complex models. For example, TRScore [35] estimates the quality protein complex models by learning from a voxelized 3D grid representation of the protein-protein interface using a deep convolutional RepVGG architecture. DOVE [50] applies a 3D convolutional neural network with voxelized representation of protein complexes, while incorporating atomic interaction types and their energetic contributions. Additionally, it integrates knowledge-based statistical potentials GOAP [116] and ITScore [147] to capture atomic interaction energies, demonstrating competitive scoring performance. Recently, representation learning with graph neural networks [51] is gaining significant attention, leading to the development of several protein complex model quality estimation methods. For example, GNN-DOVE [39] uses a graph attention network [44] by embedding protein complex interfaces as graphs. DPROQ [148] uses a gated-graph transformer model (GTN) for complex quality estimation. Despite the progress, these methods do not consider two key factors that can significantly improve protein-protein interface quality estimation performance. First, the geometry of the interaction interface often carries key information for protein complexes [53, 149], but none of the protein complex model scoring methods incorporate multimeric interaction geometries, including the inter-atomic distance and orientations of the residues at the interaction interface. Second, most of the protein complex model scoring approaches typically rely on the physicochemical properties or energetic contributions of the interacting atoms but do not consider the availability of evolutionarily information in the form of multiple sequence alignments (MSAs). That is, they ignore the quality of MSAs during scoring.

Here, we present a protein–protein interface quality estimation method called PIQLE by deep graph learning of multimeric interaction geometries. PIQLE formulates protein– protein interface quality estimation as a graph learning task by constructing a graph considering the residues at the interaction interface and estimates the interface quality by training a multi-head GAT using sequence- and structure-derived node features along with evolutionarily information and newly introduced edge features in the form of inter-atomic interaction

distance and orientations capturing multimeric interaction geometries. Unlike the existing GNN-based methods operating on voxelized representation of the protein–protein interface to estimate the overall interface quality, PIQLE first estimates the quality of the individual interactions between the interfacial residues by edge-level error regression and then probabilistically combines the estimated quality of the interfacial residues for scoring the overall interface. Large-scale benchmarking on multiple widely used protein docking decoy sets demonstrates that PIQLE consistently attains better performance than existing complex model quality estimation methods in terms of various evaluation measures including hit rate, success rate, reproducibility of model-native similarity scores and distinguishability between acceptable and incorrect models. By conducting rigorous ablation study and comparison with the self-assessment module of AlphaFold-Multimer repurposed for protein complex scoring on an independent dataset, we directly verify that the improved performance of our method is connected to the effectiveness of the multi-head GAT in leveraging multimeric interaction geometries and evolutionary information along with the other sequence- and structure-derived features. PIQLE is freely available at https://github.com/Bhattacharya-Lab/PIQLE.

5.3 Materials and methods

Figure 5.1 illustrates our protein-protein interface quality estimation framework consisting of a graph representation of the interaction interface, featurization including multimeric interaction geometries, and quality estimation of individual interacting residues by edge-level error regression using multihead graph attention network followed by probabilistic combination for the estimation of the overall interface quality.

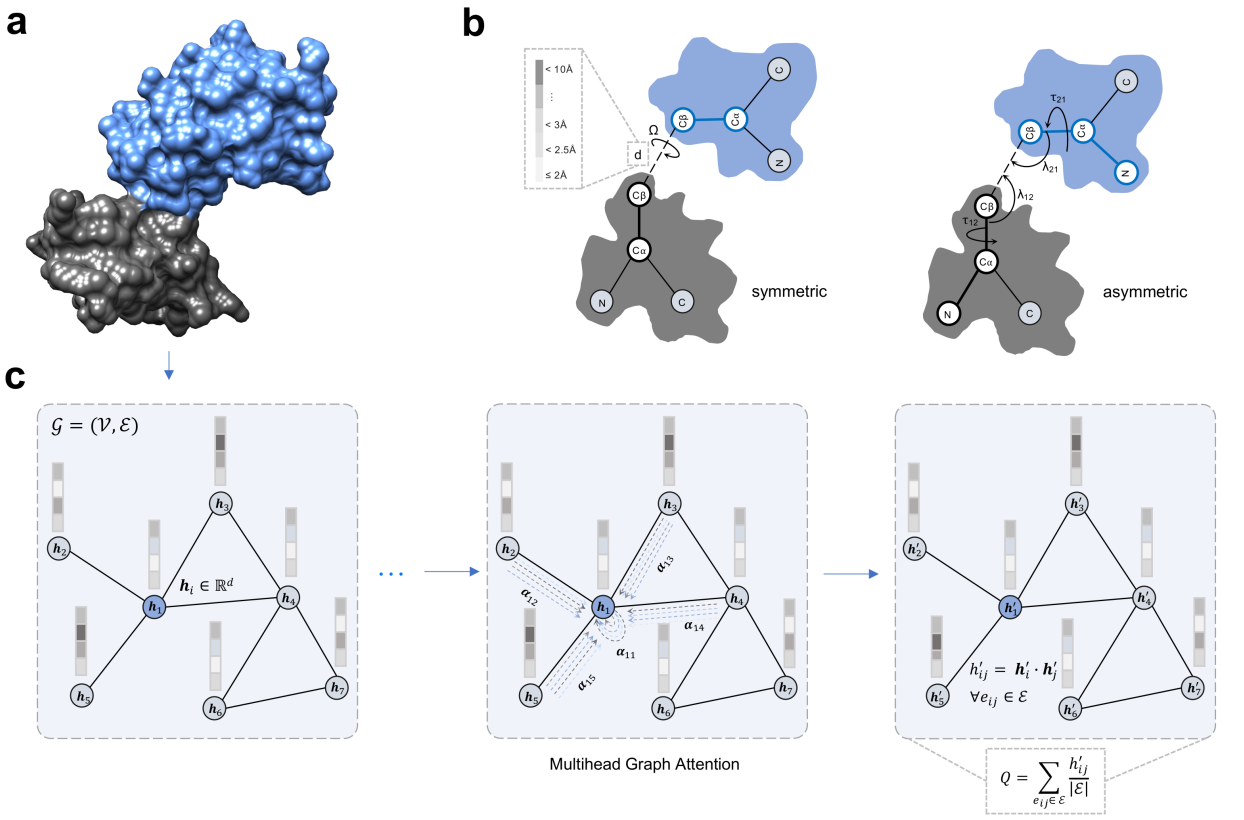

Figure 5.1: Illustration of the PIQLE framework for protein-protein interface quality estimation. (a) The input predicted protein complex structure with its two interacting monomers colored in grey and blue. (b) Multimeric interaction geometries characterized by the interatomic distance and orientations of the residues at the interaction interface. (c) Graph representation of the interaction interface and quality estimation of individual interacting residues by edge-level error regression using multihead graph attention network followed by a probabilistic combination.

5.3.1 Graph representation and featurization

We represent the protein-protein interface as a graph $G = (V, E)$, in which a node $v \epsilon V$ represents an interface residue and an edge $e \epsilon E$ represents an interacting interface residue pair. We consider an interface residue pair to be interacting if their $C\beta$ atoms ($C\alpha$ for glycine) are within 10Å [150]. With such a graph representation, we use a total of 17 node features and 27 edge features describing each interface residue and their interactions including sequence- and structure-based node features and multimeric interaction geometric edge features. We describe them below.

Node features

Residue encoding: We cluster 20 naturally occurring amino acids into 4 classes including polar, nonpolar, positively charged, and negatively charged (Supplementary Table 6.1) [151, 152, 153]. For a given residue belonging to one of these classes, we perform one-hot encoding of the residue using 5 class bins with the last bin reserved for the non-standard amino acids belonging to none of the 4 preceding bins, leading to 5 features for each node in the interfacial graph (i.e., a binary vector of 5 entries).

Relative residue positioning: To capture the relative positional information for each residue, we extract 1 feature for each node in the interfacial graph corresponding to each of the amino acid residues in a sequence as follows:

$$relPos(aa) = \frac{aa^n}{L} \tag{5.1}$$

where aa^n is the position of the n^{th} residue in the sequence and L is the length of the sequence.

Secondary structure and solvent accessibility: We use DSSP [104] program to calculate the secondary structure and solvent accessibility from the structure. We transform 8-state secondary structures into 3-state by grouping them into helices, strands, and coils for each of the residues in the sequence (Supplementary Table 5.1). Additionally, we discretize the real-valued solvent accessibility into two states of buried and exposed using the solvent-accessible surface area for the corresponding residue (Supplementary Table 6.1). We then use the one-hot encoding of 3-state secondary structures and 2-state solvent accessibility, resulting in 5 features.

Local backbone geometry: We calculate phi (ϕ) and psi (ψ) backbone torsion angles from the structure to capture the local backbone geometry of each residue. We perform sinusoidal

and cosine transformations of the angles to account for the periodicity of the angles [154], leading to 4 features.

Evolutionarily information: We compute the number of effective sequences (N_{eff}) from the MSAs of the individual monomer and concatenated MSA of the complex to account for the depth of the MSAs, thereby considering the availability of evolutionarily information. To generate MSAs from an individual monomeric sequence, we run HHBlits [155] for three iterations with an E-value inclusion threshold of 10^3 for searching against the Uniclust30 [69] database with a query sequence coverage of 10% and maximum pairwise sequence identity of 90%. [156]. We then calculate the normalized number of sequences as:

$$Neff_{eff}^{norm} = \frac{N_{eff}}{\sqrt{L}} \tag{5.2}$$

where N_{eff} is the reciprocated sum of the number of sequences in the MSA having a sequence identity greater than 80% to the N^{th} sequence and L is the length of the sequence [65]. We calculate the normalized number of effective sequencesN_{eff} following the approach described in DeepMSA [72] Additionally, we concatenate both of the MSAs of the individual monomeric sequences using GLINTER [157]following the method described in ComplexContact [158] and compute N_{eff} of the concatenated MSA using the same aforementioned strategy. Therefore, each of the nodes in a graph has 2 evolutionarily features including the N_{eff} computed from the MSA of its corresponding monomeric sequence and the N_{eff} calculated for the concatenated MSA. The two evolutionarily features are then considered as global features for all the nodes in the interface graph.

Edge features

Multimeric interaction distance: To capture multimeric interaction geometry, we discretize the Euclidian distance between the C atoms (C for glycine) of the interacting interface

residue pairs into 17 bins ranging from 2Å to 10Å having a bin width of 0.5Å. The discretized interaction distance is represented by one-hot encoding, resulting in 17 edge features.

Multimeric interaction orientation: In addition to $C\beta$-$C\beta$ distances, we also include the orientations of the interacting interface residue pairs by extending the work of trRosetta [109] for multimers. In particular, our multimeric interaction orientation is represented by 3 torsion ($\Omega, \tau_{12}, \tau_{21}$) and 2 planar angles ($\lambda_{12}$, λ_{21}), as shown in Fig. 5.1b. The Ω torsion angle measures rotation along the virtual axis connecting the C atoms of the interacting interface residue pairs, and $\tau 12$, λ_{12} ($\tau 21$, λ_{21}) angles specify the direction of the C atom of interface residue from the first (second) interacting monomer in a reference frame centered on the interface residue from the second (first) interacting monomer. Unlike the symmetric torsion angle Ω, θ and λ are asymmetric and depend on the order of the monomeric interacting interface residue pairs. Once again, we perform sinusoidal and cosine transformations of the angles, leading to 10 features.

5.3.2 Network architecture

Figure 5c shows the architecture of our multi-head GAT for protein–protein interface quality estimation. The network consists of four multi-head graph attention layers [44]. All the intermediate layers have four attention heads except for the output layer, which has one attention head. We perform hyperparameter selection on an independent validation set using grid search to determine the optimal number of layers and heads (Supplementary Table 5.2). The input layer of the network takes the interfacial graph G consisting of nodes V and edges E with the associated nodes and edge features as $G(V_i \epsilon \mathbb{R}^{v \times 1} \times E_{ij} \epsilon \mathbb{R}^{e \times 1})$. We use an empirically selected hidden dimension of 32 for the input layer with a scaling factor of 0.5 for each succeeding layer. Additionally, we perform a concatenation operation of all the heads along the output dimension of 1. Therefore, the output dimension of each intermediate layer depends on the number of hidden dimensions, and thus the output dimension of the

multi-head attention layer l is as follows:

$$X^l = \iint_{k=1}^{4} h^l \tag{5.3}$$

where k represents the number of heads and h is the hidden dimension at layer l. Of note, each multi-head attention layer performs a series of operations before it feeds the output to the next layer. First, we concatenate both the node and edge features and perform a linear transformation to embed both the node (hli) and edge (e_{ij}^l) input features with the initialized weight W to dimensional hidden features assigned to each node [44, 159] as follows:

$$z^l = W^l h_i^l e_{ij}^l \tag{5.4}$$

where z represents the embedded features at layer l and W is the learnable network parameters, normalized using the Xavier weight initialization procedure at each layer to prevent vanishing and exploding gradient problems [160]. We then compute an attention score a_{ij} between the neighboring nodes of each edge by performing self-attention on the incident nodes as:

$$a_{ij}^l = \sigma\left(W(z_i^l | z_j^l)\right) \tag{5.5}$$

where z_i^l and z_j^l are the embeddings of the incident nodes of an edge. Both embeddings are concatenated, and a dot product is computed with a learnable weight vector $W(W \epsilon \mathbb{R}^D)$ where D represents the input dimension. Meanwhile, the node features of each node are updated with the combination of neighboring node features and the attention score a_{ij} as follows:

$$h_i^l = \sigma\left(\sum_{j \epsilon N(i)} a_{ij}^l z_j^l\right) \tag{5.6}$$

5.4 Model training

For the assignment of the ground truth interface quality score during training, we first calculate the observed $C\beta$–$C\beta$ distance between the interacting interface residue pairs in the predicted complex structural model (d_{ij}^{model}) and the corresponding residue pairs in the native structure (d_{ij}^{native}). We then assign a normalized ground truth interface quality score z_{ij} to the edge e_{ij} as follows:

$$z_{ij} = \begin{cases} 1 & if\, d_{ij}^{model} < 10\, and\, d_{ij}^{native} < 10 \\ \frac{1}{1+\left(\frac{|d_{ij}^{model} - d_{ij}^{native}|}{}\right)^2} & otherwise \end{cases} \tag{5.7}$$

where $|d_{ij}^{model}$ - $d_{ij}^{nativel}|$ is the observed edge-level error between the interacting interface residue pairs corresponding to the edge eij, and d_0 is a normalizing constant whose value is set to 10Å to be consistent with the 10Å threshold used for defining interacting residue pairs in the literature [150].

During model training, we learn the interface quality score z_{ij} for each edge e_{ij} through edge-level error regression by optimizing the mean squared error loss function with sum reduction using the Deep Graph Library [161]. We use the Adam optimizer [83] with a learning rate of 0.001 and a weight decay of 0.0005. The training process consists of at most 500 epochs on an NVIDIA A40 GPU having an early stopping criterion with patience set to 40 to prevent overfitting.

5.4.1 Estimation of protein-protein interface quality

During the inference, we first estimate the interface quality score for each of the interacting residue pairs in the interface graph through edge-level error regression by computing the dot product between the predicted embeddings of the corresponding nodes as follows:

$$h^{'}_{ij} = h^{i}_{j}.h^{'}_{j} \tag{5.8}$$

where, $h^{'}$ and h^{i}_{j} represent the node embeddings of nodes i and j connected by the edge e_{ij} in the final layer of the multi-head graph attention network. We then probabilistically combine the estimated quality scores of the individual interfacial residue pairs for estimating the overall interface quality score Q as follows:

$$Q = \sum_{e_{ij}\epsilon\varepsilon} \frac{h^{'}}{|\varepsilon|} \tag{5.9}$$

where, || represents the number of interfacial residue pairs in the model and $_{ij}$ is the estimated interaction score for the edge e_{ij}. The overall interface quality score Q ranges between 0 and 1 with a higher score indicating better protein-protein interface quality.

5.4.2 Datasets

To train the graph attention network of PIQLE, we use the docking benchmark set of Dockground [162] version 2 (hereafter called Dockground v2) containing 179 dimeric protein complex targets having the length ranging from 92 to 894 residues with 100 complex structural models for each target, generated by docking the unbound structure of the receptor to the ligand.

To benchmark our method, we use the docking benchmark set of Dockground version 1 (hereafter called Dockground v1), comprising 61 dimeric protein complex targets having lengths ranging from 107 and 892 residues. We discard all targets from the Dockground v1 test dataset overlapping with our training set Dockground v2 using an average pairwise sequence identity cutoff of 20%, resulting in 23 dimer targets with an average of 109 decoys per target. Additionally, we use the Heterodimer-AF2 (hereafter called HAF2) [163] dataset consisting of 13 targets having an average of 105 decoys per target with the length ranging from 78 to 1248 residues generated using AlphaFold-Multimer [138].

For ablation studies, we use the docking Benchmark version 4.0 [164] comprising 69 dimeric protein complex targets having the length ranging from 23 to 822 residues that are nonoverlapping to the targets in the training and benchmarking datasets (pairwise sequence identity cutoff of 20%) with each target having 100 complex structural models generated by ZDOCK [140]. It is worth noting that all the datasets used for training, benchmarking, and ablation studies are non-overlapping with an average pairwise sequence identity of less than 20% between any pair of datasets (Supplementary Table 5.3).

5.4.3 Evaluation metrics and competing methods

We assess the performance of our method using various evaluation metrics based on the DockQ score [165] as the ground truth. DockQ incorporates various CAPRI measures including F_{Nat}, LRMS, and iRMS to evaluate the quality of protein-protein docking models [166]. FNat is defined by the fraction of native interfacial contacts in the model. The root mean square deviations (RMS) for the ligand (LRMS) and interface (iRMS) between the model and the target is calculated as follows:

$$RMS_scaled(RMS, d_i) = \frac{1}{1 + \left(\frac{RMS}{d_i}\right)^2} \tag{5.10}$$

Where d_i represents the scaling factors: d1 for LRSM and d2 for iRMS. d1 and d2 are optimized to be 8.5Å and 1.5Å respectively, based on the ability to separate models according to CAPRI classifications in terms of F1 scores [165, 166]. Finally, the DockQ score is calculated by combining all the above-mentioned scoring factors as follows,

$$DockQ(FnatLRMS, iRMS, d1, d2) = \frac{(F_{nat} + RMS_{scaled}(LRMS, d1) + RMS_{scaled}(iRMS, d2))}{3} \tag{5.11}$$

Where the DockQ score ranges between 0 and 1 with a higher score indicating better model quality. Overall, the continuous DockQ measure is able to rank and score docking models at a more detailed atomic level of structural granularity. We use three evaluation criteria to measure the protein-protein interface quality estimation performance: (i) ability to reproduce the ground truth DockQ scores, (ii) ability to rank complex structural models, and (iii) ability to distinguish acceptable from incorrect models. For the first criterion, we use the Spearman correlation coefficient (ρ) between the estimated quality of the protein complexes and their corresponding DockQ scores. Consequently, a higher correlation indicates better reproducibility. For the second criterion, we use the top-N success rate (herein: SR) and top-N hit rate (herein: HR) [35]. The top-N success rate is calculated as the percentage of complex targets having at least one acceptable model among top N ranked models as follows:

$$SR(N) = \frac{S(N)}{K} \times 100 \tag{5.12}$$

where, $S(N)$ is the number of complex targets having at least one acceptable model among top N ranked models and K is the total number of targets, where a cutoff of DockQ = 0.23

is used to identify acceptable models [137]. The top-N hit rate is calculated as the fraction of acceptable models among top-ranked models relative to all acceptable models in the entire dataset as follows:

$$HR(N) = \frac{H(N)}{K} \times 100 \tag{5.13}$$

where, H(N) is the total number of acceptable models among top N ranked models and M is the total number of acceptable models in the dataset. Higher success and hit rate indicate better ranking ability, especially for low values of N. We evaluate the success and hit rate of topranked models for various values of N including top-1, top-5, top-10, top- 15, top-20, top-25, and top-30. We further evaluate the methods' ranking performance on the top-N ranked models after categorizing them into acceptable, medium, and high-quality complex models based on their DockQ scores using CAPRI criteria. For the third criterion, we perform receiver operating characteristics (ROC) analysis using a DockQ score cutoff of 0.23 to separate acceptable and incorrect models. Meanwhile, the Area Under the ROC curve (AUC) quantifies the ability of a method to distinguish between acceptable and incorrect models with a higher AUC indicating better distinguishing ability.

We compare the performance of PIQLE against a number of existing protein complex quality estimation methods ranging from physics-based approaches to machine learning-based methods. As a representative physics-based method, we compare PIQLE against ZRANK2 [146], which is an improved version of ZRANK [145]. For a fair comparison, we use a min-max normalization strategy to scale ZRANK2 energy scores to the same range as the predicted scores of other methods including PIQLE as follows:

$$ZRANK2 = \frac{X - X_{max}}{X_min - X_{max}} \tag{5.14}$$

where, X is the raw ZRANK2 estimated energy scores and X_{min} and X_{max} represent the smallest and largest estimated scores, respectively, considering all predicted complex structural models for a specific target.

We also compare the performance of PIQLE against various machine learning-based approaches including 3D Convolutional Neural Network (3DCNN)-based methods TRScore [28] and four variants of DOVE [50]: DOVE-Atom20, DOVEAtom40, DOVE-GOAP, and DOVE-Atom40+GOAP as well as recent graph neural network-based methods GNN-DOVE [39] and DProQA [163]. We exclude the comparison to GNNDOVE and TRScore on the Dockgorund v1 test dataset due to the overlap of the training datasets used in GNN-DOVE and TRScore with the complex targets present in the Dockground v1 dataset. Meanwhile, all methods are included for performance comparison on the HAF2 test dataset.

5.5 Results

5.5.1 Reproducing ground truth DockQ scores

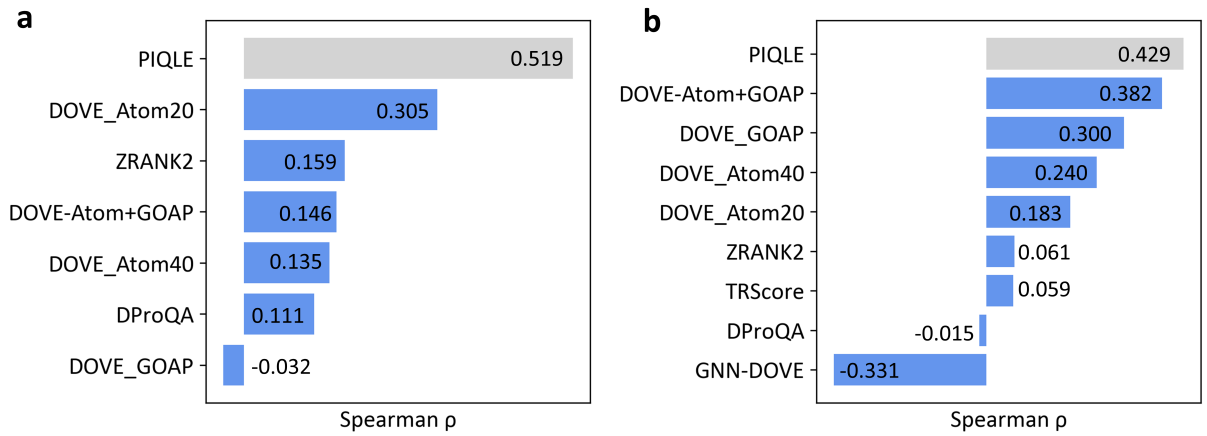

Figure 5.2: Reproducibility of ground truth DockQ scores for PIQLE (in gray) and the competing methods (blue), sorted in decreasing order of Spearman correlations coefficient (ρ) between the estimated qualities of the protein-protein interfaces and their corresponding DockQ scores on (a) Dockground v1 and (b) HAF2 datasets.

Figure 5.2 shows the Spearman correlation coefficients (ρ) between the estimated qualities of the protein-protein interfaces and their corresponding ground truth DockQ scores for PIQLE and the other competing methods. PIQLE consistently outperforms all other competing methods in both Dockground v1 and HAF2 datasets. On Dockground v1, PIQLE attains the highest Spearman correlation of 0.519 which is much better than the second-best 3D Convolutional Neural Network (3DCNN)- based method DOVE-Atom20 (0.305), the recent graph transformer network (GTN)-based method DProQA (0.111), and physics-based scoring function ZRANK2 (0.159). The same trend continues for the HAF2 dataset, in which PIQLE attains the highest Spearman correlation of 0.429 which is significantly better than the other competing methods. 3DCNN -based method DOVE remains the second-best method with its variant DOVE-Atom+GOAP attaining a Spearman correlation of 0.382. The recent graph neural network-based methods DProQA and GNNDOVE, however, fail to generalize on the HAF2 dataset attaining negative Spearman correlations of -0.015 and -0.331, respectively. In summary, our method PIQLE delivers an improved ability to reproduce ground truth DockQ scores with high fidelity.

5.5.2 Ranking complex structural models

Figure 5.3a-5.3b show the complex model ranking performance of PIQLE and the other competing methods in terms of the success rate (SR) metric, which evaluates the ability of a method to select at least one acceptable model within top N ranked models. As shown in Fig. 5a, PIQLE consistently achieves the highest SR among all methods for almost all top-N rankings in the Dockground v1 dataset. The noticeably higher SR of PIQLE at low values of N such as top-1 ($\sim$44%) and top-5 ($\sim$74%) is particularly noteworthy. In the HAF2 dataset (Figure 4.3b), PIQLE attains a higher top-1 SR of $\sim$77of the other methods such as ZRANK2 and DOVE-ATOM20 achieve comparable or higher SR values, particularly for high values of N. Overall, PIQLE frequently attains higher success rates, particularly when

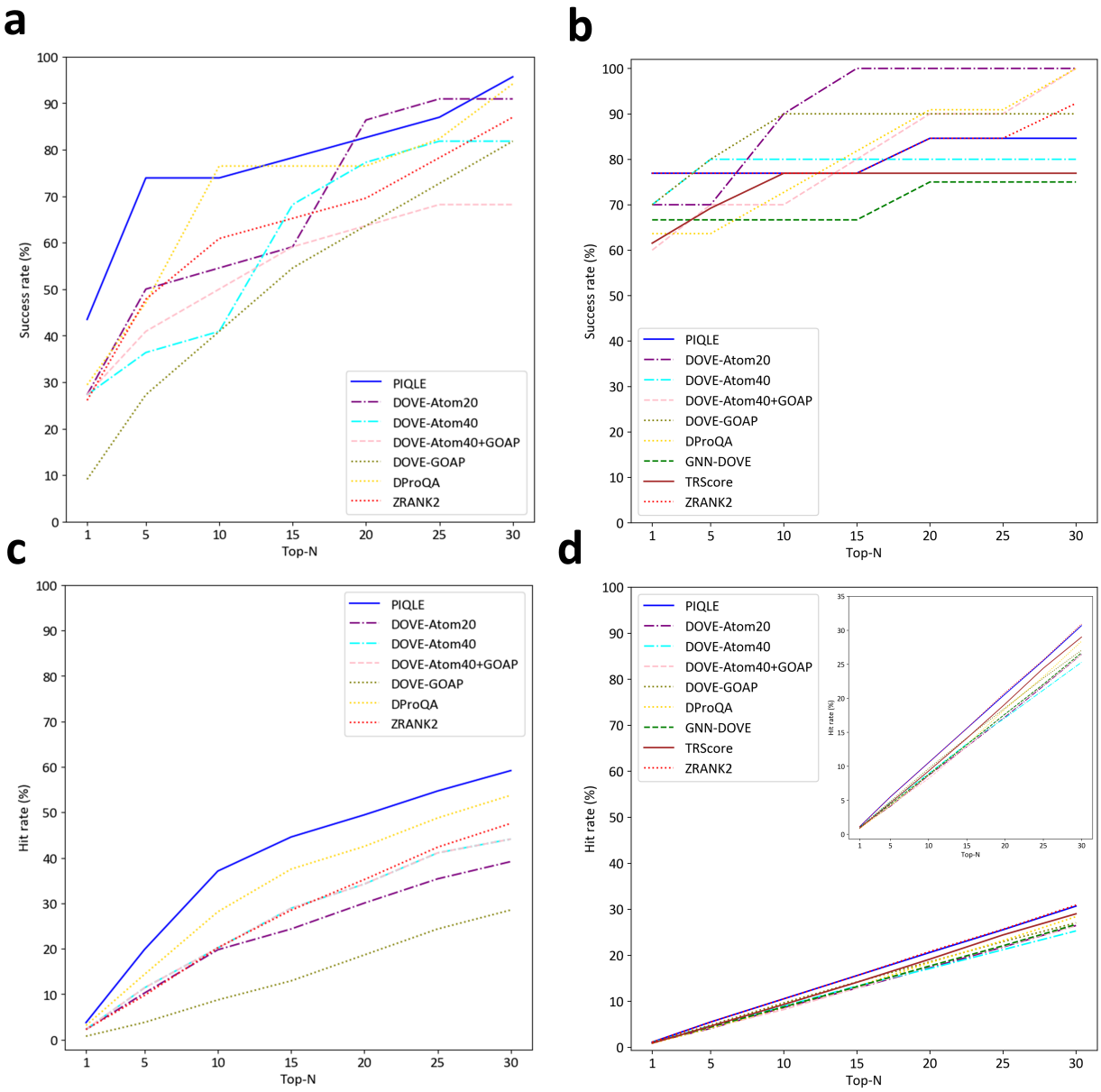

Figure 5.3: Ranking complex structural models for PIQLE and the competing methods in terms of success rate on (a) Dockground v1 dataset, (b) HAF2 dataset and hit rate on (c) Dockground v1 dataset, (d) HAF2 dataset based on top-1, top-5, top-10, top-15, top-20, top-25, and top-30 models. A cutoff of DockQ = 0.23 is used to identify acceptable models.

N is low.

Figure 5.3c-5.3d show the ranking ability of PIQLE and the other competing methods in terms of the hit rate (HR) metric, which evaluates the performance of a method based on the total number of acceptable models among top-ranked models relative to all acceptable

models in the entire dataset. As shown in Figure 5.3c, PIQLE significantly outperforms all other competing methods by achieving the highest HR on Dockground v1 dataset, for all values of N. For example, PIQLE improves the top-10 HR by more than 30% (37.079 vs 28.125) over the second-best method DProQA. PIQLE also consistently attains better HR performance on the HAF2 dataset as shown in Fig. 5d. It is interesting to note the somewhat low HR performance of all methods including ours on the HAF2 dataset. While all methods, particularly the machine learning-based approaches achieve higher SR on the HAF2 dataset, they appear to be less effective at selecting a large proportion of acceptable models from a smaller number of top-ranked models measured by HR, suggesting a need for further improvement. Nevertheless, our new method PIQLE strikes an ideal balance to deliver top performance in terms of both success rate and hit rate across different datasets, indicating its all-round ability in ranking complex structural models.

We further benchmark the ranking performance of PIQLE, and other competing methods on the same test datasets based on the CAPRI classifications of acceptable, medium, and high quality complex models in terms of DockQ scores (Supplementary Figures 6.7-6.9). Overall, PIQLE delivers reasonable ranking performance across different Top-N success and hit rates for acceptable, medium, and high quality complex models.

5.5.3 Distinguishing acceptable from incorrect models

In addition to reproducing the ground truth DockQ scores with high fidelity and accurately ranking complex structural models, the ability to distinguish acceptable from non-acceptable prediction is critically important. Figure 5.4 shows the area under the ROC curve (AUC) attained by PIQLE and the competing methods on the test datasets. PIQLE consistently outperforms all other competing methods by achieving the best AUC on both the Dockground v1 and HAF2 datasets. For the Dockground v1 dataset, AUC attained by PIQLE is 0.711 which is closely followed by the second-best performing method DOVE-Atom40+GOAP

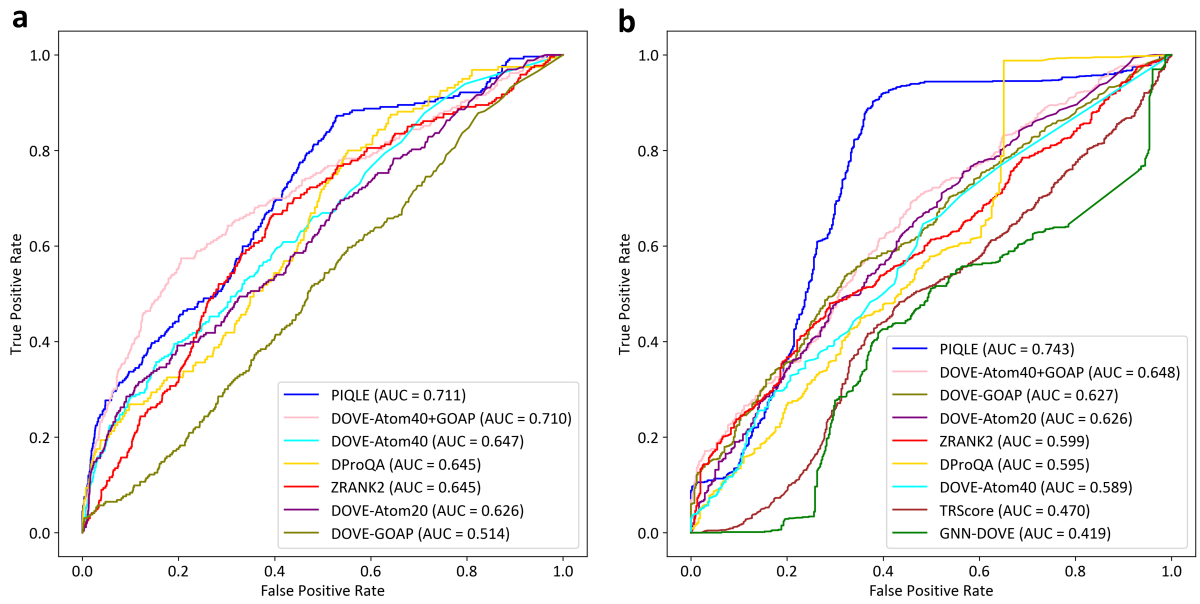

Figure 5.4: Distinguishability of acceptable vs. incorrect models for PIQLE and the competing methods on (a) Dockground v1 and (b) HAF2 datasets. A cutoff of DockQ = 0.23 is used to separate acceptable from incorrect models.

with an AUC of 0.710. On the HAF2 dataset, PIQLE attains an AUC of 0. 743, which is significantly higher than all competing methods including the second-best method DOVE-Atom40+GOAP having an AUC of 0.648. In summary, PIQLE exhibits an improved ability to distinguish between acceptable and non-acceptable prediction.

5.6 Case study

Figure 5.5 shows some representative examples of protein-protein interface quality estimation by PIQLE for selected targets from Dockground v1 and HAF2 datasets with varying degrees of predictive modeling accuracy. For a reasonably well-predicted complex structural model for Dockground v1 target 1r0r having a DockQ score of 0.732 (Fig. 5a), PIQLE estimates an interfacial quality score of 0.625. Apart from a few false positive interacting residue pairs, most of the interface regions in this predicted complex structural model are correct with an F1 score of 0.674 considering the previously defined C -C distance threshold of 10Å [150]

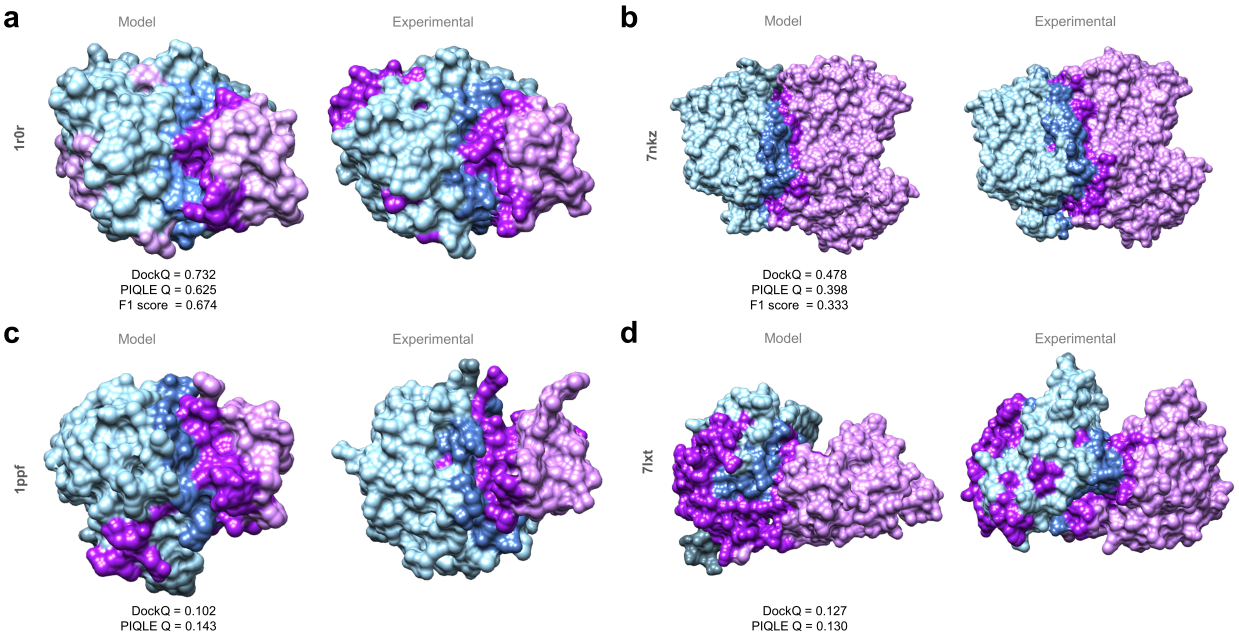

Figure 5.5: Case study on protein-protein interface quality estimation by PIQLE using predicted complex structural models for (a) Dockground v1 target 1r0r, (b) HAF2 target 7nkz, (c) Dockground v1 target 1ppf, (d) HAF2 target 7lxt. For each target, the interacting protein chains are colored in blue (chain 1) and purple (chain 2) with the interface regions highlighted in darker shades of blue and purple. The experimental structures for each target are shown side-by-side with the observed interface regions annotated.

for identifying the true interacting residue pairs. For a moderate quality predicted complex structural model for HAF2 target 7nkz having a DockQ score of 0.478 and several false positive interacting residue pairs with an F1 score of 0.333 (Figure 5.5b), PIQLE estimates a moderate interfacial quality score of 0.398. Additionally, Figure 5.5c-5.5d show two low-quality predicted complex structural models for Dockground v1 target 1ppf and HAF2 target 7lxt having a DockQ score of 0.102 and 0.127, respectively, with noticeably wrong interfaces. For these models, PIQLE estimates much lower interfacial quality scores of 0.143 and 0.130, respectively.

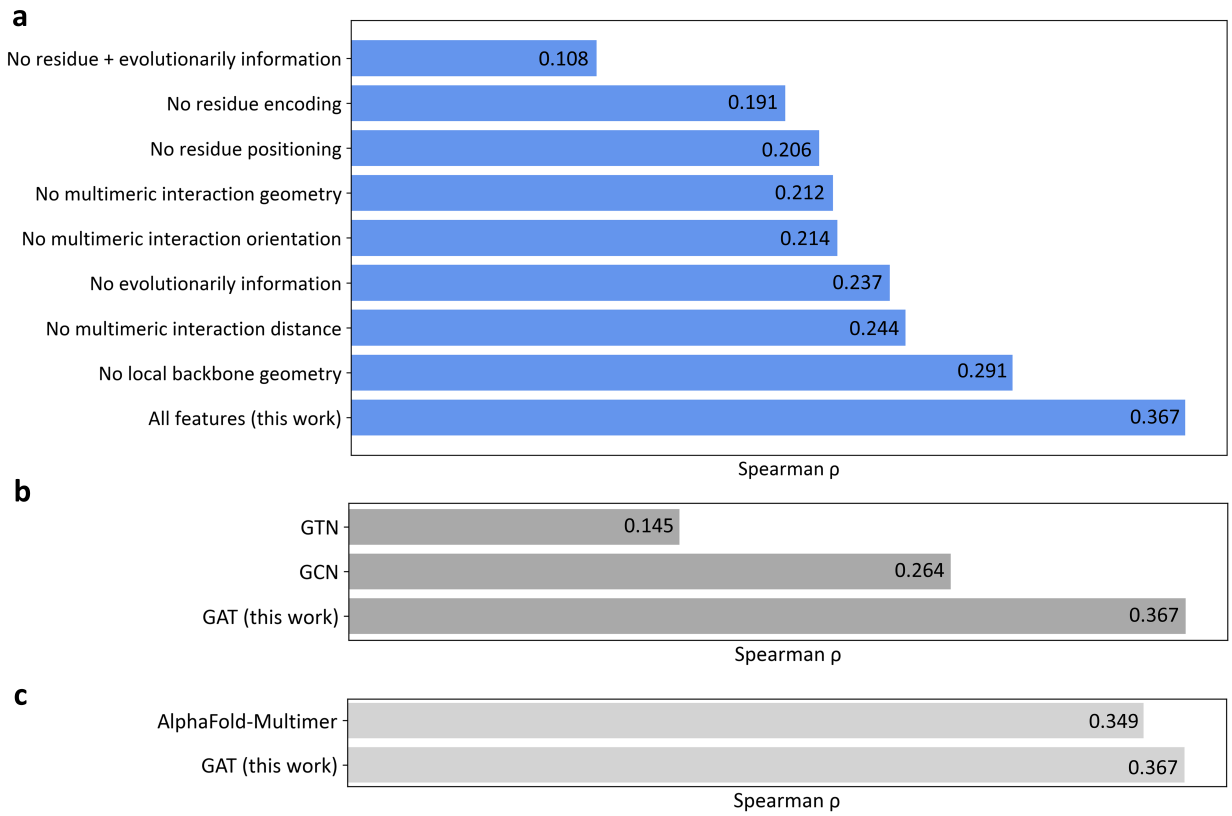

Figure 5.6: Ablation study on the independent ZDOCK validation dataset in terms of Spearman correlations coefficient (ρ) between the estimated qualities of the protein-protein interfaces and their corresponding DockQ scores by (a) gradually isolating individual feature or groups of features during model training, (b) training two baseline graph neural network models employing graph convolutional network (GCN) and graph transformer network (GTN) architectures, and (c) the performance comparison between graph attention network (GAT) in PIQLE and the ipTM scores by AlphaFold-Multimer.

5.7 Ablation study

To examine the relative importance of the features adopted in PIQLE, we conduct feature ablation experiments by gradually isolating the contribution of individual feature or groups of features during model training and evaluating the accuracy on the independent ZDOCK validation dataset. Figure 5.6a shows the Spearman correlation coefficients (ρ) between the estimated qualities of the protein-protein interfaces and their corresponding ground truth DockQ scores when various features are isolated from the full-fledged version of PIQLE. The results demonstrate that all features contribute to the overall performance achieved by

PIQLE. For example, we notice an accuracy decline when we isolate the sequence-based features one by one including amino acid residue encoding (No residue encoding) and relative residue positioning (No relative residue positioning). Importantly, discarding evolutionarily information noticeably declines the overall performance (No evolutionarily information), indicating the effectiveness of MSA-derived evolutionarily information. Not surprisingly, we notice a dramatic performance drop when both the residue-based features and the evolutionary information are isolated (No residue + evolutionarily information). Similarly, we also notice a performance drop when the feature based on local backbone geometry is discarded (No local backbone geometry). Additionally, we notice a consistent accuracy decline when we discard the newly introduced edge features based on multimeric interaction distance (No multimeric interaction distance), multimeric orientation (No multimeric interaction orientation), and their combination (No multimeric interaction geometry). That is, the improved performance of our method is connected to the effective integration of multimeric interaction geometries.

To further investigate the contribution of the multi-head graph attention network model used in PIQLE, we train two baseline graph neural network (GNN)-based models for protein-protein interface quality estimation: graph convolutional network (GCN) [43] and graph transformer network (GTN) [45]. All baseline networks are trained on the same training dataset using the same set of input features as the full-fledged version of PIQLE. Following the same approach as used for PIQLE's graph attention network (GAT), we perform hyperparameter selection for graph convolutional network (GCN) and graph transformer network (GTN) on the same independent validation set using grid search to determine the optimal number of layers and heads for GTN and the optimal number of layers for GCN (Supplementary Table 6.2). Figure 6.6b shows the performance of PIQLE compared to the baseline networks on the independent ZDOCK validation dataset in terms of the Spearman correlation coefficients (ρ) between the estimated qualities of the protein-protein interfaces and their corresponding ground truth DockQ scores. The multihead graph attention network

architecture of PIQLE significantly outperforms the other baseline networks, demonstrating its effectiveness for protein-protein interface quality estimation task.

We further compare the performance of PIQLE with the interface predicted TM scores (ipTM) predicted by the selfassessment module of AlphaFold-Multimer [138] repurposed for protein complex scoring (Roney and Ovchinnikov, 2022). It is important to note that ipTM is a self-assessment score generated by AlphaFold-Multimer for estimating the accuracy of their own predicted complex structural models in terms of the quality of the multimeric interaction interface. As such, the ipTM scores predicted by AlphaFoldMultimer are not equivalent to the interface quality scores estimated from an independent protein complex scoring method, such as PIQLE. Nonetheless, the comparison may offer some interesting insights. We utilize an extended version of the AF2Rank method [167] repurposed for protein complex scoring based on the self-assessment module of AlphaFold-Multimer, freely available as a Google Colab Notebook at https://colab.research.google.com/github/sokrypton/ColabDesign/blob/main/af/examples/AF2Rank.ipynb as of February 26, 2023, for generating the ipTM scores. Figure 4.6c shows the performance in terms of Spearman correlation coefficient (ρ) on the ZDOCK set for the estimated interface quality scores by PIQLE and the ipTM scores predicted by the selfassessment module of AlphaFold-Multimer repurposed for protein complex scoring. PIQLE convincingly outperforms ($\rho = 0.367$) the repurposed self-assessment complex scoring of AlphaFold-Multimer ($\rho = 0.349$), even though the feature ablated variants of PIQLE (Fig. 5a) as well as the baseline GNNs GTN and GCN (Figure 5.6b) fall short. The results further demonstrate the contribution of both the network architecture and features used in PIQLE for improved protein–protein interface quality estimation performance beyond what is attainable by the self-assessment module of AlphaFold-Multimer repurposed for protein complex scoring.

5.8 Conclusion

This work introduces PIQLE, a new method for protein–protein interface quality estimation by deep graph learning of multimeric interaction geometries. PIQLE exploits multi-head GAT architecture leveraging multimeric interaction geometries and evolutionarily information along with sequence- and structure-derived features to estimate the quality of the individual interactions between the interfacial residues and then probabilistically combines the estimated quality of the interfacial residues for scoring the overall interface. We demonstrate that PIQLE attains state-of-the-art protein–protein interface quality estimation performance by conducting large-scale benchmarking on multiple widely used protein docking decoy sets. Our ablation study and comparison with the self-assessment module of AlphaFold-Multimer repurposed for protein complex scoring on an independent validation set confirm the contribution of various features adopted in PIQLE and the effectiveness of the multi-head GAT architecture.

Our study leads to a number of future directions to consider: of particular interest is the possibility of broadening the applicability of our method for higher order oligomers and large protein assemblies. Further, a promising direction for future work is to consider the diversity of predictive modeling ensemble and conformational states of the interacting monomers for interface quality estimation for interacting proteins having multi-state conformational dynamics. Finally, integrating complementary features, such as residue-level self-assessment confidence estimates for the interacting protein chains and sequence-based disorder prediction coupled with a richer deep graph representation learning framework may further boost protein–protein interface quality estimation performance. We expect our method to be extended to other biomolecular interface characterization, including estimating the quality of predicted protein interaction with other molecules, such as DNA, RNA and small ligands.

5.9 Data availability

The raw data used in this study, including the datasets for train, test, and validation are collected from publicly available sources. The Dockground v2 training set is available at https://dockground.compbio.ku.edu/downloads/unbound/decoy/decoysset2-1.0.tgz. The Dockground v1 test set is available at https://dockground.compbio.ku.edu/downloads/unbound/decoy/decoys1.0.ziphttps://dockground.compbio.ku.edu/downloads/unbound/decoy/decoys1.0.zip. The Heterodimer-AlphaFold2 test set is available at https://zenodo.org/record/6569837/files/DproQ_benchmark.tgz. The ZDOCK docking benchmark version 4.0 validation set is available at https://zenodo.org/record/6569837/files/DproQ_benchmark.tgz.

5.10 Funding

This work has been partially supported by the National Institute of General Medical Sciences [R35GM138146 to D.B.] and the National Science Foundation [DBI2208679 to D.B.].
Conflict of Interest: none declared.

5.11 Supplementary Information

Table 5.1: Categorization of several node features adopted in PIQLE

	Properties	Category
5-state amino acid encoding	ALA VAL LEU ILE PRO PHE MET TRP	Hydrophobic
	GLY SER THR CYS TYR ASN GLN	Hydrophilic
	LYS ARG HIS	Basic
	ASP GLU	Acidic
	Any other residues	Neutral/None
3-state secondary structure	G, H, I	H
	E, B	E
	Other	C
2-state solvent accessibility	RSA (ACC / Max. SA) <25%	B
	RSA (ACC / Max. SA) >25%	E

Table 5.2: Hyperparameter searches for graph attention network (GAT), Graph Transformer Network (GTN), and Graph Convolutional Network (GCN) on the independent ZDOCK validation dataset in terms of Spearman correlations coefficient (ρ) between the estimated protein-protein interface quality scores and their corresponding DockQ scores.

Hyperparameter		GAT	GTN	GCN
# Layers	# Heads	Spearman	Spearman	Spearman
4	2	0.168	0.116	0.219
	4	0.367	0.048	
	6	0.199	0.042	
	8	0.266	0.067	
6	2	0.155	0.042	0.264
	4	0.178	0.015	
	6	0.300	0.074	
	8	0.288	0.122	
8	2	0.222	0.039	0.171
	4	0.280	0.085	
	6	0.325	0.041	
	8	0.289	0.053	
10	2	0.200	0.071	0.214
	4	0.189	0.056	
	6	0.172	0.108	
	8	0.185	0.040	
12	2	0.173	0.099	0.209
	4	0.218	0.121	
	6	0.230	0.070	
	8	0.227	0.044	
14	2	0.220	0.127	0.193
	4	0.199	0.036	
	6	0.202	0.095	
	8	0.235	0.056	
16	2	0.139	0.145	0.240
	4	0.238	0.041	
	6	-0.020	0.037	
	8	0.225	0.042	

Table 5.3: Pairwise sequence identity between training, testing, and validation datasets.

Datasets	Dockground v1[a]	Dockground v2[b]	ZDOCK[c]	HAF2[d]
Dockground v1		0.183	0.192	0.162
Dockground v2	0.183		0.179	0.152
ZDOCK	0.192	0.179		0.133
HAF2	0.162	0.152	0.133	

[a,d]testing dataset
[b]training dataset
[c]validation dataset
[d]Heterodimer-AF2 dataset

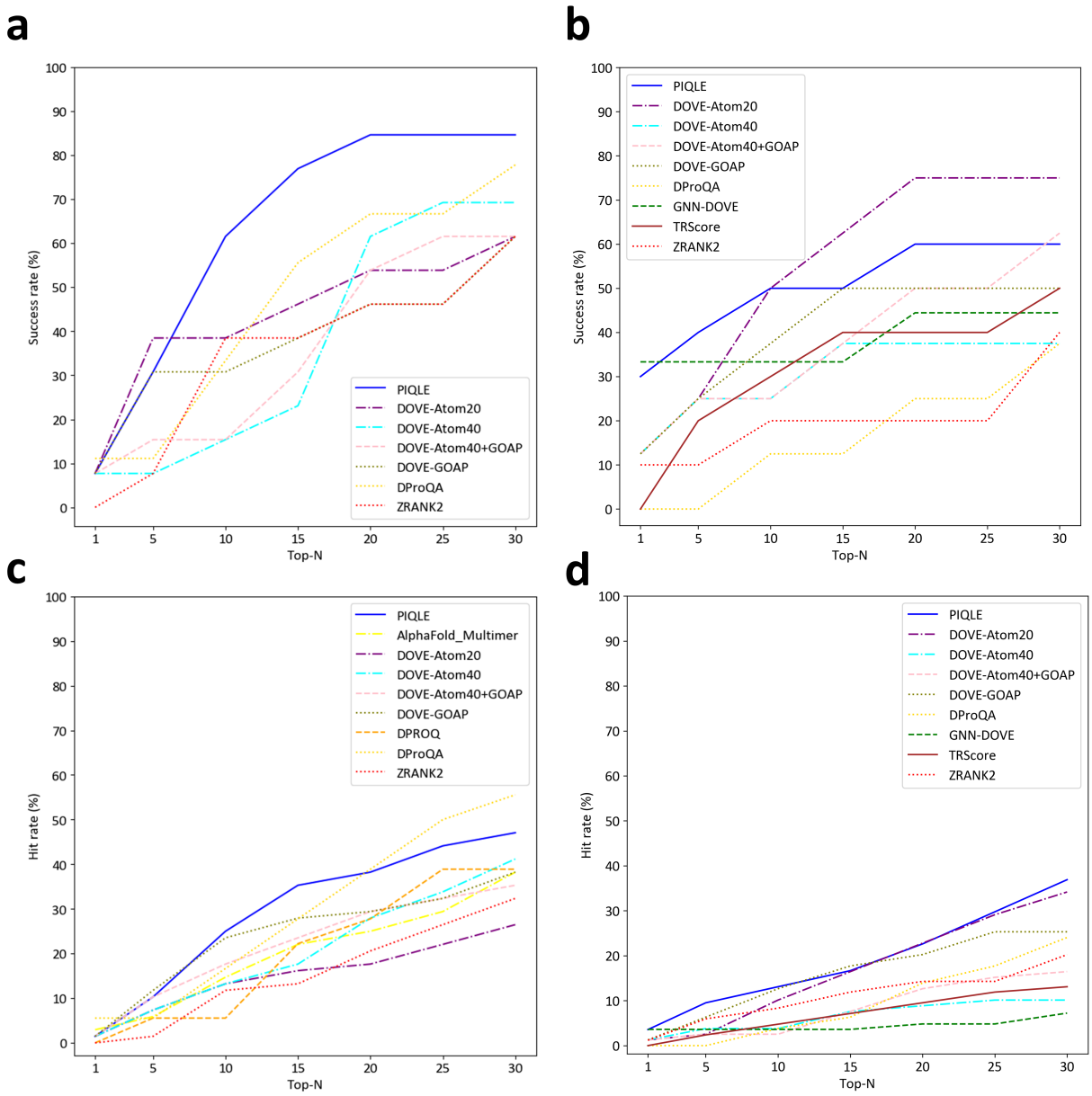

Figure 5.7: Ranking complex structural models with acceptable quality for PIQLE and the competing methods in terms of success rate on (a) Dockground v1 dataset, (b) HAF2 dataset and hit rate on (c) Dockground v1 dataset, (d) HAF2 dataset based on top-1, top-5, top-10, top-15, top-20, top-25, and top-30 models. Models having DockQ scores ranging between 0.23 and 0.49 are considered acceptable quality models.

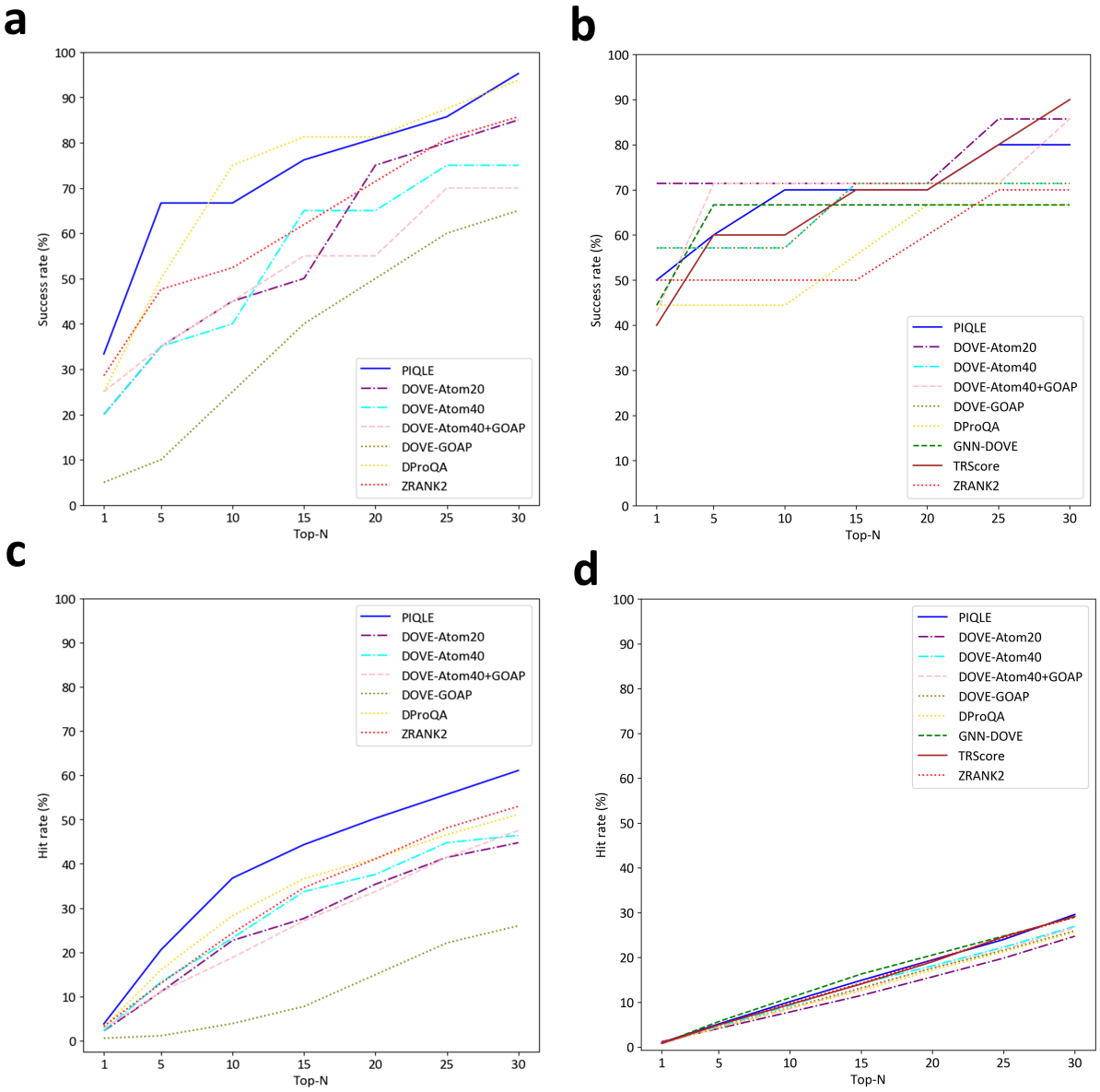

Figure 5.8: Ranking complex structural models with medium quality for PIQLE and the competing methods in terms of success rate on (a) Dockground v1 dataset, (b) HAF2 dataset and hit rate on (c) Dockground v1 dataset, (d) HAF2 dataset based on top-1, top-5, top-10, top-15, top-20, top-25, and top-30 models. Models having DockQ scores ranging between 0.49 and 0.80 are considered medium quality models.

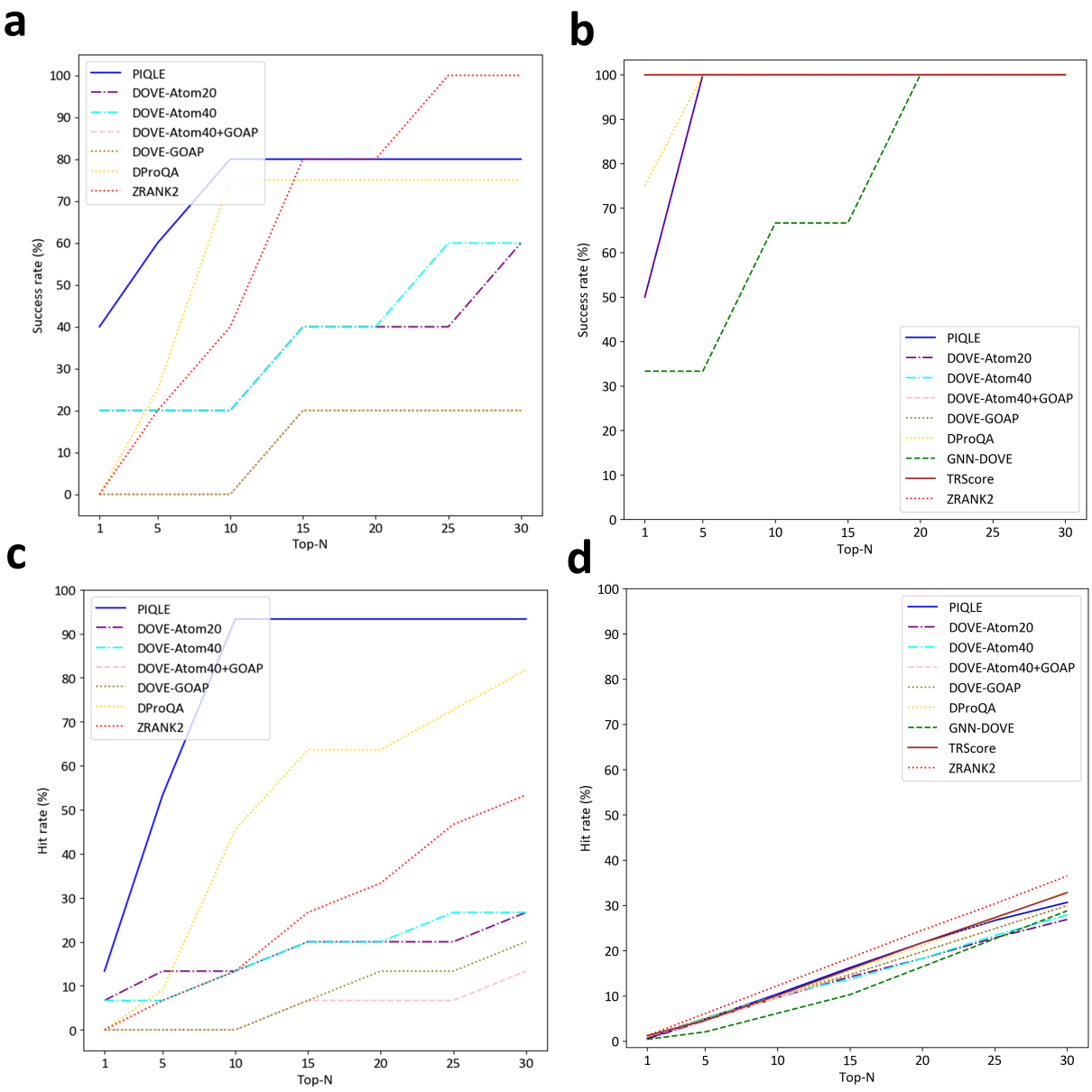

Figure 5.9: Ranking complex structural models with high quality for PIQLE and the competing methods in terms of success rate on (a) Dockground v1 dataset, (b) HAF2 dataset and hit rate on (c) Dockground v1 dataset, (d) HAF2 dataset based on top-1, top-5, top-10, top-15, top-20, top-25, and top-30 models. Models having minimum DockQ scores of 0.80 are considered high-quality models.

Chapter 6

EquiRank: improved protein-protein interface quality estimation using protein-language-model-informed equivariant graph neural networks

6.1 Abstract

Motivation: Quality estimation of the predicted interaction interface of protein complex models helps in guiding the highly accurate protein complex structure prediction. Despite recent progress fueled by symmetry-aware deep learning architectures and large-scale Protein Language Models (pLMs), existing methods for estimating protein complex quality have yet to fully exploit the collective potentials of these advances for the accurate estimation of protein-protein interface.

Results: We present EquiRank, an improved protein-protein interface quality estimation method by leveraging the strength of a deeper symmetry-aware Equivariant Graph Neural Network (EGNN) and integrating the large-scale Protein Language Model-based ESM-2 embeddings. Our method estimates the quality of the protein-protein interface through an effective graph-based representation of interacting residue pairs, incorporating diverse representative features, including ESM-2 embeddings, and then by learning the representation

using a symmetry-aware EGNN. Our experi-mental results demonstrate improved Ranking performance on diverse datasets over our recent method PIQLE and several other protein complex quality estimation methods including the self-assessment module of AlphaFold-Multimer repurposed for protein complex scoring and VoroIF_GNN across different performance evaluation metrics. Additionally, our ablation studies demonstrate the contributions of both the Protein Language Model and the equivariance property of EGNN to improved protein-protein interface quality estimation performance. Availability: EquiRank is freely available at https://github.com/mhshuvo1/EquiRank

6.2 Introduction

Protein-protein interactions play a fundamental role in driving a wide range of biological processes [134, 168, 169]. Although there has been a significant leap toward solving the monomeric protein models by AlphaFold2 [2], the prediction of their interactions still remains challenging [136, 137, 138]. Existing protein complex modeling methods typically generate alternative complex models (a.k.a. decoys) by performing conformational sampling of the protein-protein interaction interfaces [139, 140, 141, 142] As such, the scoring of protein-protein interfaces for the identification of the most accurate conformation is, therefore, an important component of a successful protein complex structure modeling process [143, 144]. There has been promising progress in the development of various methods for estimating the accuracy scores of protein complexes with the application of various machine learning models. For instance, TRScore uses the ResNet-inspired [170] VGG network to learn the voxelized representation of protein complexes for scoring protein complexes. Similarly, DOVE [39] employs Convolutional Neural Network (CNN) [34] to score protein complexes, utilizing 3D voxelized representation integrated with atomic energy. Recent advances in Graph Neural Network (GNN)-based models have gained attention for their ability to effectively represent molecular structures [51, 171] thus prompting their application to protein complex quality

estimation problems. For instance, in our recent work, the PIQLE [27] method successfully applies the Graph Attention Network [44] to estimate the accuracy of the protein-protein interface. VoroIF_GNN [29] estimates the accuracy of protein complexes leveraging attention-based Graph Neural Network to learn atom-level graph derived from the Voronoi tessellation-based interface representations. DProQA [163] employs a graph transformer network for estimating the quality of protein complexes. GDockScore [40] utilizes a bi-directional graph attention network for estimating the quality score of protein complexes. GNN-DOVE [39] represents the protein interface and graph and employs Graph Attention Network (GAT) to estimate the protein complex score. In addition to exploiting graphical representations of protein complexes with several representative features, new protein complex quality estimation methods leverage protein language models (PLMs), which have revolutionized diverse predictive tasks, including protein structure prediction, protein function prediction, and protein engineering prediction, in recent years [10, 172, 173, 174, 175, 176, 177] For instance, DeepRank-GNN-esm uses a graph neural network (GNN) that incorporates embeddings from the transformer-based protein language model ESM-2 [10] to better capture the graph representation of a protein complex. Despite the success of existing protein complex quality estimation methods in applying GNN-based approaches, such as Graph Attention Network (GAT) and Graph Transformer Network (GTN), operating efficiently on protein graphs lacking a fixed order and providing important invariant properties regardless of node permutation, they do not inherently capture symmetry under certain rotations and translations while dealing with such 3D objects such as proteins [46, 178]. Therefore, considering the inherent 3D structure of proteins, Equivariant Neural Network (EGNN) [46] is desirable that incorporates the spatial location of each residue, providing properties such as equivariance to rotation and translation, while also maintaining invariance to node permutation. Furthermore, the incorporation of embeddings from the Protein Language Model ESM-2, trained on large-scale protein sequences, for a more comprehensive representation of nodes, combined with EGNN, may enhance the overall robustness of the framework [179]. Here

we present, an improved protein-protein interface quality estimation method EquiRank with the application of a Symmetry-aware Equivariant Graph Neural Network (EGNN). Starting from a given protein complex, EquiRank first extracts the interface graph of interacting residue pairs. Subsequently, the nodes corresponding to residues and the edges corresponding to their interactions are represented using various representative features including the embeddings from the protein language model based ESM-2 model and our novel multimeric geometries, as employed in our recently published protein-protein interface quality estimation method PIQLE [27]. In addition to delivering improved quality estimation performance for protein-protein interface over PIQLE, EquiRank demonstrates better performance than contemporary state-of-the-art protein complex quality estimation methods across diverse datasets and various accuracy measures. The observed improved performance as validated through our ablation studies is directly connected to the integration of Protein Language Models-based embeddings and the efficient equivariance properties of EGNN. EquiRank is freely available at https://github.com/mhshuvo1/EquiRank.

6.3 Method

6.3.1 Features generation

Interface graph representation: We represent a protein complex as an interface graph G = (V, E) consisting of interacting residue pairs with each of the residues in the interface as a node v ∈ V and an interacting residue pair as an edge e ∈ E as shown in Figure 6A. We extract the interface residue pairs from a given protein complex based on the inter-residue distances of their Cβ (Cα for glycine) atoms. Specifically, we define a residue pair as interactive when the distance between their Cβ (Cα for glycine) atoms is less than 10Å [27]. We represent each of the nodes in the interface graph with 349 features including protein language model (pLM)-based and AlphaFold2 distilled Multiple Sequence Alignment (MSA) features. Additionally,

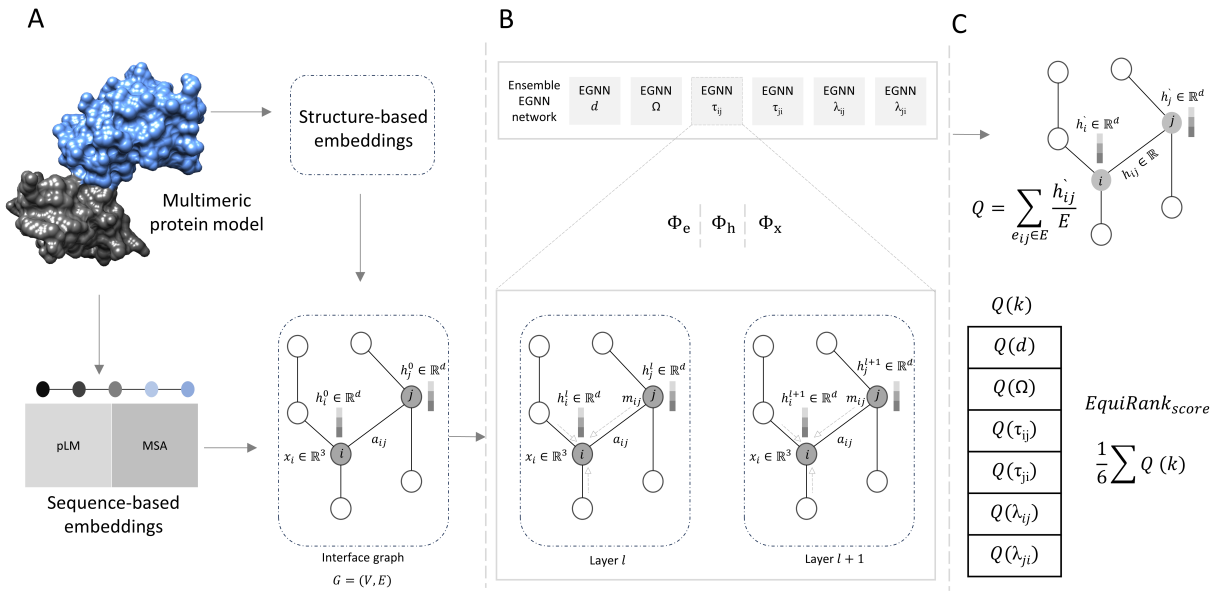

Figure 6.1: Flowchart of EquiRank framework for protein-protein interface quality estimation. A) Generation of sequence- and structure-based features, including pLM-based sequence embeddings and MSA-based encoding, and multimeric distance and orientation, respectively, from the predicted protein complex structure with two interacting monomers colored in blue and gray. B) Architecture of ensemble Equivariant Graph Neural Network (EGNN). C) Edge-level regression of interacting residue pairs for transformed distance (d) and normalized angular RMSD of multimeric orientations Ω, λ_{12}, λ_{21}, τ_{21}, τ_{21}, and their probabilistic combination for estimating the protein-protein interface quality.

we represent each of the edges with multimeric distance and orientation having 29 features.

Node feature: We generate a total of 349 features including both sequence- and structure-based features for representing each of the nodes in the interface graph.

1. Protein Language Model (pLM) based features (33): We use the pre-trained ESM-2 model [10] having 15B parameters to generate the sequence embeddings having a shape of L $\times$ 33, where L is the length of the sequence, for each of the amino acid sequences of interacting monomer in the protein complex. Afterward, we perform a sigmoidal transformation on the resulting embeddings to generate 33 Protein Language Model (pLM)-based features.

2. Multiple sequence alignment features (256): We employ ColabFold [180] to generate

Multiple Sequence Alignment (MSA) using MMseq2 [71] for each of the amino acid sequences in the protein complex. The generated MSA is then input to the EvoFormer blocks of AlphaFold2 [2], implemented in ColabFold, producing distilled MSA representations encoded as a dictionary. Subsequently, we extract the first row of the distilled MSA representation ("msa_first_row" from the dictionary) and then apply a sigmoidal transformation to generate 256 MSA features.

3. Evolutionarily features (2): We generate evolutionarily features in the form of the number of effective sequences (Neff) computed from the Multiple Sequence Alignment (MSA) of the individual amino acid sequences within the protein complex and its concatenated MSA, by following a similar methodology as adopted in PIQLE [146]. The number of effective sequences (Neff) represents the depth of the MSA thereby considering the evolutionarily information.

4. Amino acid encoding (21): We represent each of the nodes in the interface graph, corresponding to a specific amino acid residue, using a one-hot encoded binary vector, consisting of 20 naturally occurring amino acid types and a gap for non-standard amino acid [65].

5. Relative residue positioning (1) We obtain the relative positional information for each node in the interface graph corresponding to each of the residues in the sequence of a model as follows,

$$\text{relPos}(aa) = \frac{\text{aa}^n}{L} \tag{6.1}$$

6. Where aa^n is the n-th residue in the sequence and L is the length of the sequence.

7. Secondary structure and solvent accessibility (13) We use the DSSP [104] program to generate secondary structure and solvent accessibility from the structure. For each of the residues corresponding to the nodes in the graph, we generate a binary vector

of one-hot encoding for 8-state secondary structure types, resulting in 8 secondary structure features. Additionally, we transform the 8-state secondary structure into a 3-state by grouping them into helices, strands, and coils [27]. Subsequently, we generate a one-hot encoded binary vector of 3-state secondary structures, resulting in 3 features. We discretize real-valued solvent accessibility into buried and exposed [146] and generate 2 features by performing one-hot encoding of the corresponding types.

8. Local backbone geometry (4): To capture the local backbone geometry, we calculate phi (ϕ) and psi (ψ) backbone torsion angles from the structure to capture the local backbone geometry of each residue. Subsequently, we generate 4 features by performing sinusoidal and cosine transformations of the angles [154].

9. Ultra shape recognition features (3): To capture the topological relationship between the residue and the overall structure, we calculate residue-level Ultra Shape Recognition (USR) [181] features. Following a similar approach as adopted in DeepUMQA [35], we compute the residue-level USR feature representing the spatial relationship information between the structure and each specific residue using three residue distance sets. Subsequently, we apply min-max transformation on all three distance sets to generate normalized USR features for each residue of a corresponding node in the interface graph, resulting in 3 USR features.

10. Residue orientation (3): To define the orientation of each amino acid residue, we derive three features for each residue, corresponding to a node in the graph. These features include two calculated as the unit vectors in the directions of $C_{\alpha_{i+1}} - C_{\alpha_i}$ and $C_{\alpha_{i-1}} - C_{\alpha_i}$, and one feature calculated as the unit vector of $C_{\beta_i} - C_{\alpha_i}$, based on the assumption of tetrahedral geometry [94].

11. Residue neighbors (1): We calculate the number of spatial neighbors of a node in the interface graph in terms of the overall structure where two residues are considered to be neighbors if the distance of their Cβ (Cα for glycine) atoms is less than 10Å. Once

again, we perform a min-max transformation to generate the normalized number of neighbors, resulting in 1 feature.

12. Rosetta centroid energy terms (12): We generate 12 Rosetta [76, 77] centroid energy terms following a similar approach as used in QDeep [21]. For each of the nodes in the interface graph, we use the sigmoidal transformation of the energy terms for the corresponding residue in the structure and use them as 12 energy-based features.

Edge features

1. Multimeric interaction distance (17): In order to capture the geometry between the interacting residue pairs, the Euclidean distance between their C_β (C_α for glycine) atoms represents the edges. We first discretized the distance from ≤ 0.2 to < 10Å into 17 bins, each having a uniform bin width of 0.5Å. Afterward, we use the one-hot encoding of the discretized bin, resulting in 17 multimeric interaction distance-based edge features.

2. Multimeric orientation (10): To capture the orientation information between the interacting residue pairs, we extend the work of trRosetta [109] for multimers to represent the edges in the interface graph with orientation features. Specifically, each edge is represented by 3 torsion angles (Ω, τ_{ij}, τ_{ji}) and 2 planar angles (λ_{ij}, λ_{ji}) [27]. The Ω torsion angle measures rotation along the virtual axis connecting the C_β atoms of the interacting interface residue pairs, and τ_{ij}, λ_{ij} (τ_{ji}, λ_{ji}) angles specify the direction of the C_β atom of the interface residue from the first (second) interacting monomer in a reference frame centered on the interface residue from the second (first) interacting monomer. Unlike the symmetric torsion angle Ω, τ and λ are asymmetric and depend on the order of the monomeric interacting interface residue pairs. Subsequently, we perform sinusoidal and cosine transformations of the angles, leading to 10 features.

3. Relative positioning of the interacting residues (2): To capture the positional informa-

tion of interacting residue pairs in the protein complex, we obtain the relative positional information for each of the interacting residue pairs and subsequently combine them to calculate the relative positioning of the interacting residue pairs as follows,

$$\text{relPos}(\text{aa}) = \left| \frac{\text{aa}_i^n}{L} - \frac{\text{aa}_j^n}{L} \right| \tag{6.2}$$

Where aa_i^n and aa_j^n are the nth interacting residues in the structure, and L is the length of the sequence.

4. Neighbors of interacting residue pairs (1): We calculate the number of spatial neighbors of each of the interacting residue pairs in terms of the overall structure where two residues are considered to be neighbors if the distance of their Cβ (Cα for glycine) atoms is less than 10Å. As aforementioned, we perform min-max transformation to generate the normalized number of neighbors for each interacting residue and subsequently use their weighted combination, resulting in 1 feature.

6.3.2 Dataset

To train our ensemble EGNN, we collect a total of 18,022 models for a non-redundant set of 1127 dimeric targets having lengths ranging from 67 to 1,375 (Supplementary Table 6.1). We first collect 14, 400 models for 1, 097 heterodimeric targets from VoroIF_GNN [29] (hereafter called VoroIF_GNN_train). We additionally incorporate 3, 622 models for 30 dimer targets from both CASP13 and CASP14 in the training set. We benchmark the performance of our method EquiRank and other competing methods on a set of 12,195 models for 284 targets having lengths ranging from 29 to 1,677 and diverse accuracy with both correct and incorrect protein complex models (Supplementary Table 6.1). We use a DockQ threshold of 0.23 to differentiate between correct and incorrect models. First, we collect 2,845 models for 235 targets from VoroIF_GNN (hereafter called VoroIF_GNN_test) having 42% correct

and 58% incorrect protein complex models. Additionally, we collected 2,500 models for 23 targets from Dockground version 1 [162] (hereafter called Dockground v1) consisting of 10.72% correct models and 89.28% incorrect complex models. Finally, we collect CASP15 datasets for protein complex targets having 6,850 decoys for 26 targets with 45.37% correct and 54.63% incorrect decoys. Additionally, for ablation studies, we collect 2,814 models for 235 targets from VoroIF_GNN (hereafter called VoroIF_GNN_validation) having a model length ranging from 96 to 1,222 (Supplementary Table 6.1). It is noteworthy that all the datasets used for training, benchmarking, and ablation studies are non-overlapping with an average pairwise sequence identity of $< 23\%$ between any pair of datasets (Supplementary Table 6.2).

6.3.3 Evaluation metrics and Competing methods:

We evaluate the performance of our method PIQLE in terms of various evaluation metrics. For all performance measures, we use the DockQ [165] score as ground truth. To evaluate the methods' ranking ability, we use per-target Spearman correlation coefficients between the decoys' predicted and the true DockQ scores. A higher correlation indicates a better ranking ability of a method. Additionally, to evaluate the methods' ability to accurately distinguish between highly accurate models with others, we report the area under the ROC curve (AUC) with a DockQ threshold of 0.80. A higher AUC value indicates better distinguishability of a method in terms of separating high-quality models. We compare the performance of EquiRank against state-of-the-art existing complex quality prediction methods using the same evaluation metric. We use several competing methods, leveraging di-verse deep learning architectures. We compare EquiRank against graph neural network-based methods VoroIF_GNN [29] PIQLE [27], DProQA [163], GNN-DOVE [39], GDockScore [40], and DeepRank-GNN-esm [41] Additionally, we compare our method with Convolutional Neural Network (CNN)-based methods such as, variants of DOVE [33] and TRScore [28].

Moreover, we compare our method with the interface-predicted TM scores (iPTM) predicted by the self-assessment module of AlpaFold2-Multimer [138]. Specifically, we utilize an extended version of the AF2Rank [167] method repurposed for estimating the protein complex quality based on the self-assessment module of AlphaFold2-multimer.

6.3.4 Network architecture

We employ a deeper Equivariant Graph Neural Network (EGNN) [46] to estimate the quality of the protein-protein interface as shown in Figure 1B. Our deep EGNN consists of four stacked convolutional layers (EGNNConv), operating on the protein-protein interface graph. It accepts both node (h_n) and edge (h_e) features, as well as the Cartesian coordinate information (h_x) of the C (Ca for glycine) atoms of the interacting residue pairs (i, j). Each EGNNConv performs a series of operations on the edge, node, and coordinate features of the interface graph, denoted by Φ_e, Φ_h, Φ_x respectively. Each operation consists of two-layer Multi-Layer Perceptrons (MLPs).

EGNNConv starts processing by performing edge embeddings for interacting residue pairs using a message-passing operation on the edges (m_{ij}) of the interacting residue pairs. This is done by applying a two-layer MLP, Φ_e, on the node features (h_n) and edge features (a_{ij}), while considering the coordinates (x_i, x_j) features as follows,

$$m_{ij} = \Phi_e(h_i^l, h_j^l, \|x_i^l - x_j^l\|^2, a_{ij}) \tag{6.3}$$

Where m_{ij} is the edge message between the interacting residue pairs i and j, and h_i^l and h_j^l are their node features at layer l. $\|x_i^l - x_j^l\|^2$ is the Euclidean distance between the Cartesian coordinates of the interacting residue pairs i and j, a fundamental operation for implementing equivariant message passing operation within EGNN, setting it apart from the traditional Graph Neural Network (GNN).

Afterward, the edge embedding m_{ij} from the previous layer is used to update the coordinates of residue i (x_i) in the interacting residue pairs in the next layer $(l+1)$. Specifically, the coordinates of i (x_i) are updated by the sum of all relative differences of interacting residue pairs weighted by the edge embedding m_{ij} as follows:

$$x_i^{(l+1)} = x_i^l + C \sum_{j \neq i} \left(x_i^l - x_j^l\right) \Phi_x(m_{ij}) \tag{6.4}$$

Where x_i^l is the coordinates of residue i in the interacting residue pairs and $(x_i^l - x_j^l)$ is the difference between their coordinates, m_{ij} is the edge embedding from the previous layer. C is a normalizing constant computed as $\frac{1}{N(i)}$ where N is the number of neighbors of residue i.

Afterward, the network aggregates all the messages from all the neighboring nodes of i to update its features as follows,

$$m_i = \sum_{j \neq i} m_{ij} \tag{6.5}$$

Finally, a non-linear transformation is applied to the aggregated message and the node feature of i in the current layer (h_i^l), producing the node embeddings with the updated node features of residue i in the next layer $(l+1)$ as follows,

$$h_i^{(l+1)} = \Phi_h(h_i^l, m_i) \tag{6.6}$$

6.3.5 Model training

To train our ensemble EGNN models, we assign the ground truth interface quality scores to each of the interacting residue pairs, representing edges e_{ij} in the interface graph. Specifically, we generate 6 sets of features by assigning ground truth multimeric geometry, including one multimeric distance and five orientation labels. We calculate the multimeric distance label by employing a similar approach as adopted in PIQLE [27], where we first calculate the

observed $C_\beta - C_\beta$ distance between the interacting interface residue pairs in the predicted complex structural model (d_{ij}^{model}) and the corresponding residue pairs in the native structure (d_{ij}^{native}). We then assign a normalized ground truth distance score $z_{ij}(d)$ to the edge e_{ij} as follows:

$$z_{ij}(d) = \begin{cases} 1 & \text{if } d_{ij}^{\text{model}} < 10\,\text{Å and } d_{ij}^{\text{native}} < 10\,\text{Å} \\ \frac{1}{1+\left(\frac{\left|d_{ij}^{\text{model}} - d_{ij}^{\text{native}}\right|}{d_0}\right)^2} & \text{otherwise} \end{cases} \tag{6.7}$$

where $\left|d_{ij}^{\text{model}} - d_{ij}^{\text{native}}\right|$ is the observed edge-level error between the interacting interface residue pairs corresponding to the edge e_{ij}, and d_0 is a normalizing constant whose value is set to 10 Å.

Additionally, to learn the orientation error, we assign multimeric orientation for each edge e_{ij} by calculating the normalized angular RMSD between torsion $(\Omega, \tau_{ij}, \tau_{ji})$ and planar angles $(\lambda_{ij}, \lambda_{ji})$ of the interacting residue pairs in the predicted complex structural model (d_{ij}^{model}), calculated by extending the work of trRosetta [109] for multimers and the corresponding residue pairs in the native structure (d_{ij}^{native}).

$$z_{ij}(\overrightarrow{a}) = \sqrt{\left(\min\left(\left|a_{ij}^{\text{native}} - a_{ij}^{\text{model}}\right|, 2\pi - \left|a_{ij}^{\text{native}} - a_{ij}^{\text{model}}\right|\right)\right)^2} \tag{6.8}$$

where $\overrightarrow{a}$ represents the torsion angles Ω, τ_{ij}, τ_{ji}, planar angles λ_{ij}, λ_{ji}. Once again, τ and Ω are asymmetric and therefore depend on the order of the residues. Afterward, we normalize the angular RMSD for the torsion angles Ω and τ as follows:

$$\text{Normalized Angular RMSD} = \frac{1}{1 + \left(\frac{z_{ij}(\overrightarrow{a})}{\frac{\pi}{4}}\right)^2} \tag{6.9}$$

where $\overrightarrow{a} \in \{\tau_{ij}, \tau_{ji}, \Omega\}$.

However, we use more stringent normalization criteria for the planar angle $z_{ij}(\lambda)$ as follows:

$$\text{Normalized Angular RMSD} = \frac{1}{1 + \left(\frac{z_{ij}(\lambda)}{\frac{\pi}{8}}\right)^2} \tag{6.10}$$

Both $z_{ij}(d)$ and $z_{ij}(\Omega, \lambda_{ij}, \lambda_{ji}, \tau_{ij}, \tau_{ji})$ range between 0 to 1, with a higher score indicating better similarity between the interacting residue pairs in the model and the corresponding residue pairs in the native structure.

Therefore, we train our ensemble EGNN models on these 6 sets of features to independently learn $z_{ij}(d)$ and $z_{ij}(\Omega, \lambda_{ij}, \lambda_{ji}, \tau_{ij}, \tau_{ji})$ through edge-level error regression by optimizing the mean squared error loss function with sum reduction using the Deep Graph Library [50]. We use the Adam optimizer [83] with a learning rate of 0.001 and a weight decay of 0.0005. The training process consists of at most 500 epochs on an NVIDIA A100 GPU, having an early stopping criterion with patience set to 40 to prevent overfitting.

6.3.6 Estimation of protein-protein interface quality:

Figure 6C shows the estimation of the protein-protein interface quality in two steps. Using each of the trained models, we first estimate the embeddings h_i^l and h_j^l for each of the interacting residue pairs i and j respectively. Afterward, we perform a dot product between the estimated embeddings (h_i^l, h_j^l) for each of the interacting residue pairs to individually estimate their quality as follows,

$$h_{ij}^{`} = h_i^l \cdot h_j^l$$

Where h_i^l and h_j^l represent the node embeddings of the interacting residue pairs i and j respectively, and $h_{ij}^{`}$ is their estimated local quality score. For the global quality score, we first perform a probabilistic combination of the local quality of all the interacting residue

pairs $|e|$ in the interface graph, estimated by each of the trained models as follows:

$$Q(k) = \frac{\sum_{e_{ij} \in E} h'_{ij}}{|E|} \quad \text{where } k \in \{z_{ij(d)}, \Omega, \lambda_{ij}, \lambda_{ji}, \tau_{ij}, \tau_{ji}\} \tag{6.11}$$

Where $|E|$ is the total number of edges in the interface graph, and Q is the global quality, ranging between 0 and 1 from each of the ensemble models. Afterward, we estimate the EquiRank_score by performing an ensemble averaging of the global quality from each of the models trained with ground truth multimeric distance, z_{ij} and orientations $\Omega, \lambda_{ij}, \lambda_{ji}, \tau_{ij}, \tau_{ji}$ as follows,

$$\text{EquiRank\_score} = \frac{Q(d) + Q(\Omega) + Q(\lambda_{ij}) + Q(\lambda_{ji}) + Q(\tau_{ij}) + Q(\tau_{ji})}{6} \tag{6.12}$$

Where EquiRank_score ranges between 0 and 1, with a higher score indicating better protein-protein interface quality.

6.4 Results

6.4.1 Ability to rank predicted models:

Figure 6.2 shows the ranking performance of EquiRank and other competing methods in terms of per-target average Spearman correlation between the methods' predicted and ground truth DockQ (Basu and Wallner, 2016) scores on VoroIF_GNN_test and Dockground v1 datasets. Our method EquiRank outperforms all other competing methods by achieving the highest per-target Spearman on both VoroIF_GNN_test and Dockground v1 datasets. On VoroIF_GNN_test set EquiRank attains the highest Spearman correlation of 0.701 which is more than 10% improvement than the second latest GNN-based complex quality estimation method VoroIF_GNN. While substantially improving the per-

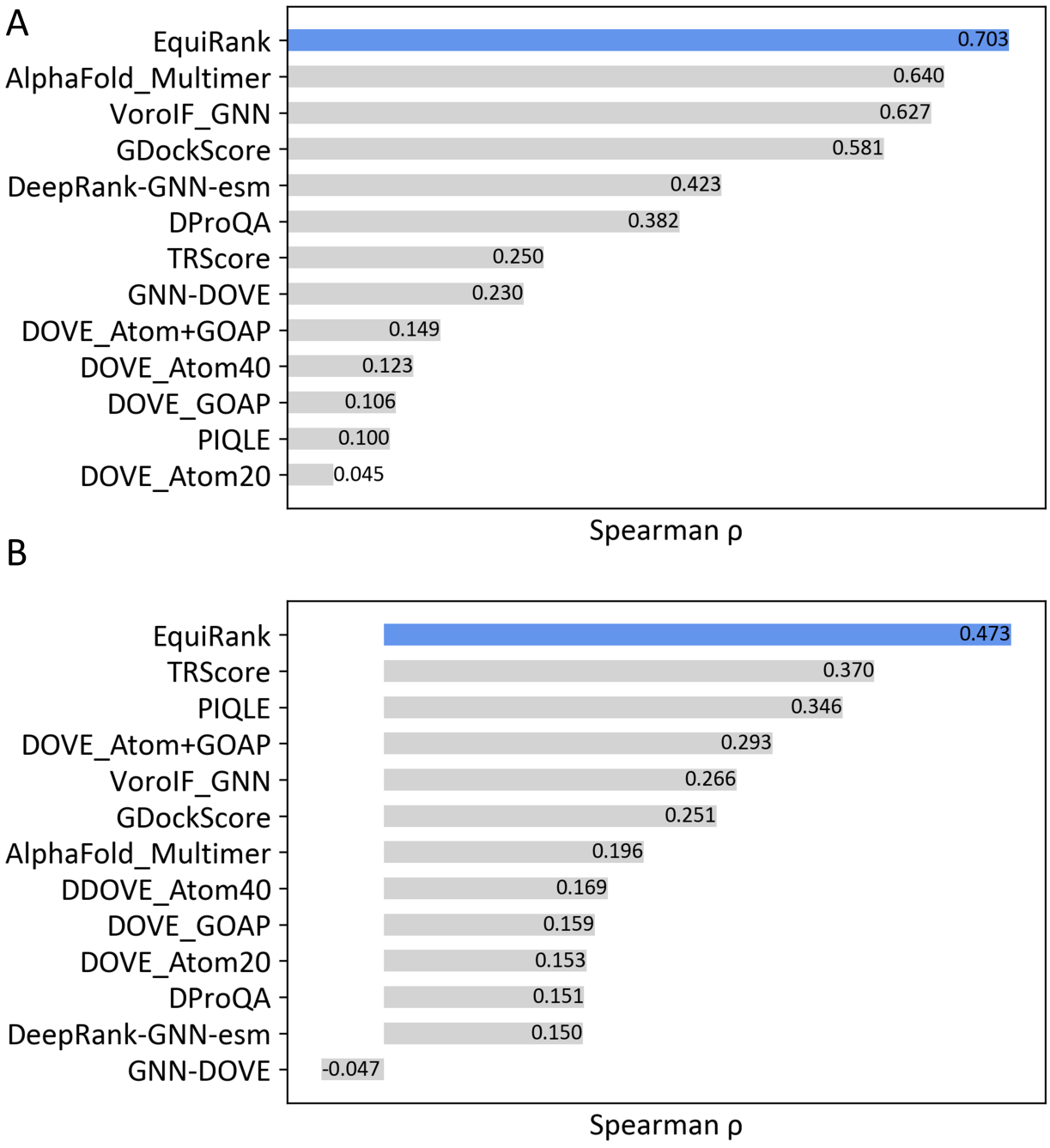

Figure 6.2: Ranking performance of EquiRank and competing methods in terms of per-target Spearman correlation coefficient (ρ) on A) VoroIF_GNN_test and B) DockGround1 datasets.

formance over our recent method PIQLE, EquiRank attains much higher Spearman correlation than any other GNN-based methods including DProQA and GNN-DOVE as well as DeepRank integrated with pLM embeddings. Additionally, on the Dockground v1 dataset, EquiRank attains the highest Spearman correlation of 0.473 than any other competing methods. AlphaFold-multimer, having the second-best per-target Spearman correlation on VoroIF_GNN dataset, attains a much lower correlation of 0.196 than EquiRank on this dataset. Although these methods demonstrate better-ranking performance on the relatively balanced VoroIF_GNN_test dataset, their suboptimal performance on the Dockground v1 dataset, which lacks a similar balance between correct and incorrect protein multimeric complexes, reveals limitations in terms of generalizability. As such, EquiRank strikes a good balance in terms of its ranking performance on both near-balance and imbalance datasets. Additionally, it is important to note that, EquiRank improves the ranking performance by almost 28% (0.473 vs 0.370) than the second-best TRscore, while improving the performance over our recent protein-protein interface quality estimation method PIQLE. Other latest GNN-based approaches including GDockScore, DPoQA, and GNN-DOVE consistently demonstrate sub-optimal performance, with GNN-DOVE having a negative per-target Spearman correlation of -0.047. Additionally, it is worth noting that EquiRank consistently demonstrates better performance than the latest pLM-based method, DeepRank-GNN-esm, indicating the collective contribution of both the Protein Language Model (pLM)-based embeddings and the symmetry-aware Equivariant Neural Network. Overall, EquiRank exhibits an improved ranking ability for a wide range of predicted protein complex models.

6.4.2 Ability to distinguish high-quality models

Figure 6.3 shows the ability of EquiRank and other competing methods method to successfully distinguish high-quality models from other models on both Dockground v1 and CASP15 benchmark datasets. As shown in Figure 6.3A, EquiRank attains the highest AUC

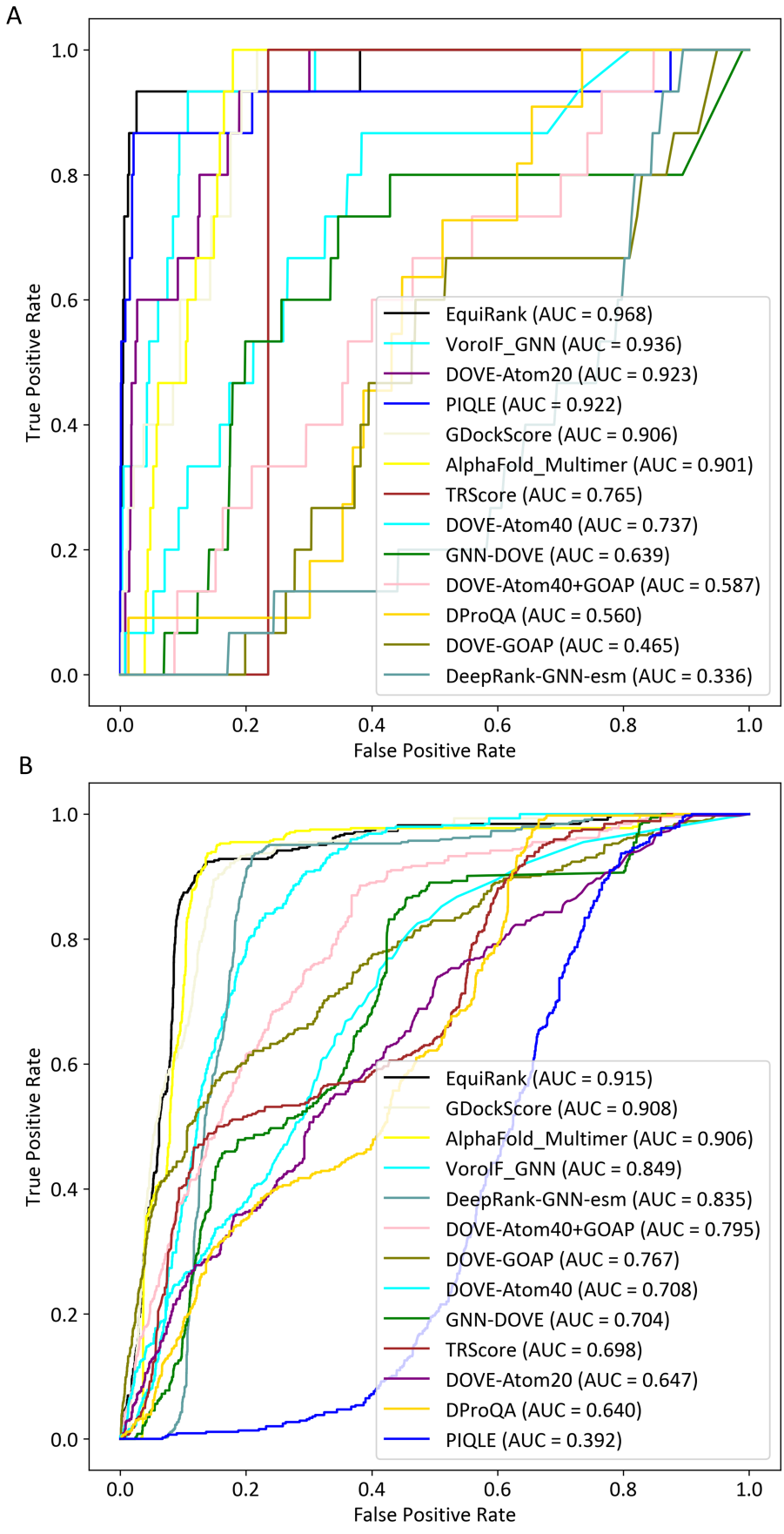

Figure 6.3: Distinguishability of high-quality models for EquiRank and other competing methods on A) Dockground v1 and B) CASP15 datasets with a DockQ threshold of 0.8.

of 0.968 on Dockground v1 dataset among all other competing methods. The GNN-based VoroIF_GNN attains the second-best AUC of 0.936. However, other GNN-based methods including GNN-DOVE, DProQA, and DeepRank-GNN-esm have limited distinguishability, having much lower AUC compared to EquiRank. On CASP15 datasets, as shown in Figure 6.3B, EquiRank once again attains the highest AUC of 0.915 which is closely followed by the second-best performing method GDockScore with an AUC of 0.908. Although VoroIF_GNN achieves the second-best AUC in Dockground v1, its performance is notably lower with an AUC of 0.849 compared to EquiRank. DeepRank-GNN-esm, utilizing Protein Language Models (pLM)-based ESM-2 embeddings, demonstrates superior performance compared to Dockground v1 (0.336 vs 0.835), yet notably lower than EquiRank. It is noteworthy to mention that the CASP15 and Dockground v1 datasets exhibit significantly different balance ratios, with CASP15 being much more balanced than Dockground v1 in terms of correct and incorrect models (Supplementary Table 6.1). While the performance of other competing methods varies when applied to different datasets with diverse model qualities, EquiRank consistently achieves better performance, demonstrating its improved ability to distinguish high-quality complex models.

6.5 Ablation study

To assess the relative importance of the features used in EquiRank for ranking performance, we conducted ablation studies. In each ablation experiment, we isolated a single feature and trained an ensemble EquiRank model. We then evaluated the ranking performance on the VoroIF_GNN_validation dataset, measuring the per-target average Spearman correlation. As shown in Figure 6.4, EquiRank with all features achieved the highest Spearman correlation of 0.662, demonstrating the contributions of all features to improved ranking performance. Importantly, the ablation of Protein Language Model (pLM)-based ESM-2 features resulted in the worst ranking performance (No pLM embeddings), with an average Spearman cor-

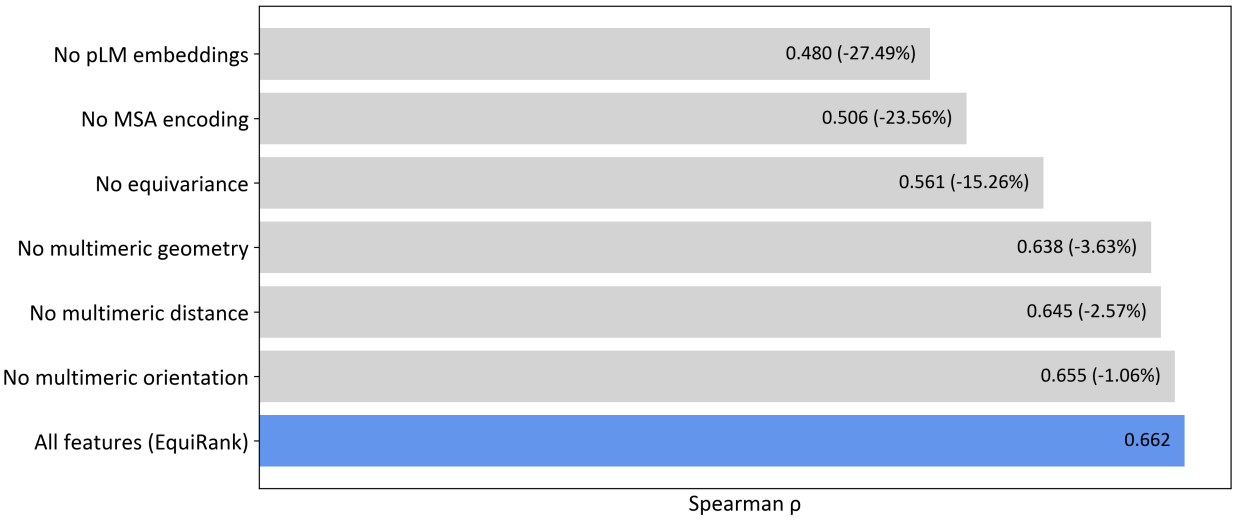

Figure 6.4: Ablation study on the independent VoroIF_GNN_validation validation dataset in terms of per-target average Spearman correlations coefficient (ρ) between the estimated qualities of the protein-protein interfaces and their corresponding DockQ scores by gradually isolating individual feature or changing network parameter model training.

relation of 0.480, indicating that the pLM contributes to improving performance by almost 28%, which is significantly larger than any other component of EquiRank. The improved MSA encoding generated by AlphaFold2 also contributes substantially to improved ranking performance (No MSA encoding), by more than 24%. Additionally, the novel representations of multimeric geometry adopted in our recent work PIQLE contribute to improved ranking performance (No multimeric geometry, No multimeric distance, No multimeric orientation), although not as much as other sequence-based features. This suggests that they can be complemented with our sequence-based representation, indicating a significant improvement over PIQLE. To examine the contribution of the equivariance properties of EGNNs in EquiRank, we disabled the equivariance update by making the model invariant. This modification resulted in a suboptimal ranking performance (No equivariance), worsening the per-target average correlation by more than 15%, from 0.662 to 0.561. This suggests the importance of maintaining equivariance properties during the learning of protein-protein interface graphs. Overall, our ablation studies demonstrate that both the representation from the Protein Language Model and the equivariant neural network have significant contributions to the

improved protein-protein interface quality estimation performance of EquiRank.

6.6 Conclusion

In this work, we present EquiRank, an improved protein-protein interface quality estimation method. EquiRank introduces several new advances over our recent protein-protein interface quality estimation method PIQLE including the application of a symmetry-aware Equivariant Graph Neural Network and integration of several new sequence- and structure-based features including Protein Language Model-based ESM-2 embeddings. EquiRank demonstrates improved quality estimation performance over PIQLE and a wide range of protein complex quality estimation methods in terms of accurately ranking protein complex models and distinguishing high-quality models on large-scale benchmarking datasets. Additionally, while existing protein complex quality estimation methods show varying performance on near-balance and relatively imbalanced datasets, EquiRank exhibits consistent and improved ability to rank and distinguish. Finally, through our ablation studies, we demonstrate that the improved performance is directly attributed to the application of equivariance properties of EGNN and the integration of new sequence-based representation including Protein Language model embeddings.

6.7 Supplementary Information

Table 6.1: Distribution of training, testing, and validation datasets

	Dataset	**Num targets**	**Num decoys**	**Correct (DockQ $\geq$ 0.23)**	**Incorrect (DockQ <0.23)**
Training	VoroIF_GNN train	1,097	14,400	42%	52%
	CASP13	20	2386	17.24%	82.76%
	CASP14	10	1329	15.95%	84.05%
Testing	VoroIF_GNN_test	235	2845	42%	58%
	Dockground v1	23	2500	10.72%	89.28%
	CASP15	26	6850	45.37%	54.63%
Validation	VoroIF_GNN validation	235	2,814	40.96%	59.14%

Table 6.2: Pairwise sequence identity between training, testing, and validation datasets.

Datasets	**VoroIF_ GNN_ train**[a]	**VoroIF_ GNN_ test**[b]	**CASP 13**[a]	**CASP 14**[a]	**CASP 15**[b]	**Dockground v1**[b]	**VoroIF_ GNN_ valdation**[c]
Voroif_ GNN_ train		19.57%	21.28%	22.28%	20.08%	19.83%	19.29%
VoroIF_ GNN_ test			21.41%	22.37%	20.16%	19.98%	19.40%
CASP 13				20.0%	17.64%	17.53%	21.35%
CASP 14					15.27%	15.09%	22.42%
CASP 15						19.53%	20.08%
Dockground v1							19.97%
Voroif_ GNN_ validation							

[a] Training datasets
[b] Testing datasets
[c] Validation datasets

Chapter 7

Scientific software development and dissemination

Here, I present a list of scientific software developed as a first and co-author of the corresponding work, along with their brief descriptions and usage. All the software mentioned below is freely available to the broader scientific community for utilization.

1. **QDeep** [21]: distance-based monomeric protein model quality estimation using deep Residual Neural Network.

   Availability: https://github.com/Bhattacharya-Lab/QDeep

   Input: protein models in PDB format, corresponding protein sequence in FASTA format, and predicted inter-residue distance information.

   Output: estimated quality score of the protein models, ranges between 0 and 1, with a higher score indicating better estimated quality.

2. **iQDeep:** [54] deep learning-based integrated protein scoring server

   Availability: http://fusion.cs.vt.edu/iQDeep

   Input: protein models in PDB format, and corresponding protein sequence in FASTA format.

   Output: estimated quality score with customized parameters, ranges between 0 and 1, with a higher score indicating better estimated quality. Visual analytics of the structural properties of the input models.

   Usage statistics: A total of 70 jobs have been processed from 10 countries.

3. **DeepRefiner:** [23] deep learning-based monomeric protein model refinement server

   Availability: http://watson.cse.eng.auburn.edu/DeepRefiner

   Input: starting structure of a protein model in PDB format.

   Outupt: refined protein structures with visual analytics of their structural properties.

   Usage statistics: processed 2,601 jobs from 71 countries

4. **PIQLE:** [27] protein-protein interface quality estimation by deep graph learning of multimeric interaction geometries.

   Availability: https://github.com/Bhattacharya-Lab/PIQLE

   Input: protein complex models in PDB format, and corresponding sequences in FASTA format.

   Output: estimated protein-protein interface quality, ranges between 0 and 1, with a higher score indicating better estimated quality.

5. **SPECS:** [182] multi-model method for improved protein scoring

   Availability: https://github.com/Bhattacharya-Lab/SPECS

   Input: protein models in PDB format and corresponding protein sequence in FASTA format.

   Output: quantitative score for the input protein models, ranges between 0 and 1, with a higher score indicating better quality.

6. **EquiPPIS:** [183] E(3) equivariant graph neural networks for robust and accurate protein–protein interaction site prediction

   Availability: https://github.com/Bhattacharya-Lab/EquiPPIS

   Input: structural- and sequence-based features for protein complex models.

   Output: predicted residue-level protein-protein interaction site.

7. **EquiPNAS:** [179] EquiPNAS: improved protein-nucleic acid binding site prediction using protein-language-model-informed equivariant deep graph neural networks.

   Availability: https://github.com/Bhattacharya-Lab/EquiPNAS

Input: structural- and sequence-based features for protein-nucleic acids complex models.

Output: prediction of protein-nucleic acids binding sites.

8. **rrQNet** [184]: deep learning method for protein contact map quality estimation by evolutionary reconciliation.

   Availability: https://github.com/Bhattacharya-Lab/rrQNet

   Input: residue-residue contact map in RR format.

   Output: estimated scores of the input contact maps

Chapter 8

Conclusion and Future work

Here we present our five research outcomes of developing data-driven methods for protein model quality estimation by leveraging the latest advances in deep learning architectures. We develop quality estimation methods for both the monomeric protein models having single chains and for the multimeric protein complex models having multiple chains/subunits. We introduce the first single-model monomeric quality estimation method that utilizes the power of an ensemble of very deep Residual Neural Networks (ResNets) while integrating novel distance-based structural representation. We further, develop a high-accuracy single-model protein quality estimation method by considering the recent breakthrough of AlphaFold2 in achieving near-experimental accuracy in protein structure prediction. We additionally, integrate the error estimation module of our quality estimation method to further improve the accuracy of monomeric protein models. We afterward, shift our focus to multimeric protein complexes due to their important role in catalyzing several biological processes. In this context, we develop multimeric protein model quality estimation methods by leveraging the strength of Graph Neural Network architectures which exhibit promise in efficiently representing and learning molecular structures, such as proteins using graph-based representation. We first develop a multimeric protein model quality estimation method by deep graph learning of multimeric geometry, showing better performance compared to the existing approaches to protein complex quality estimation methods. We afterward start developing an improved multimeric protein model quality evaluation method by exploiting recent symmetry-aware Equivariant Graph Neural Network (EGNN) with an integration of Protein Language Model-based embeddings. Our methods have been rigorously tested

and compared against the state-of-the-art approaches in multiple rounds of Critical Assessment of Techniques for Protein Structure Prediction (CASP) experiments including diverse benchmark sets of both monomeric and multimeric protein models. Overall, our methods effectively assess the quality of computationally predicted monomeric and multimeric protein models, providing accessible tools for the broader scientific community to evaluate their reliability when experimentally determined protein structures are unavailable.

In the near future, we plan to incorporate several advances to improve the performance of the latest development of our multimeric protein model quality estimation methods over its current state. Although EquiRank demonstrates better ranking ability, one of the major limitations of EquiRank is the lack of ability to reproduce the ground truth DockQ scores. Specifically, EquiRank presents an improved performance in reproducing the relative ranking of the predicted protein complex models, however, we observe relatively higher differences between its estimated scores and the ground truth DockQ score [165]. Therefore, our immediate plan is to optimize the Equivariant Neural Network architecture within the EquiRank framework to improve its representational ability to capture the sensitivity of the models' ground truth scores. In that context, we will carefully calibrate the hyperparameters of the underlying ensemble networks to observe their response during inference, ultimately to obtain an optimal set of parameters.

Secondly, we plan to incorporate additional evaluation metrics to evaluate the method's performance in terms of providing complementary benefits. To evaluate the ability of EquiRank and other competing methods to reproduce the actual ground truth DockQ scores, we will incorporate Mean Absolute Error (MAE) as an additional evaluation metric, with lower MAE indicating better performance. Mean Absolute Error (MAE) is calculated as the average absolute difference between the estimated score of a method and the corresponding ground truth DockQ scores as follows,

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|p_i - t_i| \tag{8.1}$$

where n is the number of predicted complex models, p_i is the estimated score of a multimeric complex model i and p_i is the corresponding ground truth DockQ scores.

Additionally, we will evaluate the performance of EquiRank and other competing methods in terms of selecting the best model from a pool of predicted multimeric protein models in terms of the top-1 DockQ loss. Top-1 DockQ loss is calculated as the difference between the true DockQ score of the top model selected by the estimated score and that of the most accurate model in the pool. Moreover, we may replace the DockQ score with the Quarternary Structure (QS) [185] score as an interface similarity measure between the predicted and experimental structures, which has been employed by the assessor of 15^{th} Critical Assessment of Protein Structure Prediction (CASP) experiments to evaluate methods' performance. QS score measures the similarity of interface region based on the shared interface contacts, having the ability to discriminate between alternative quaternary structures and binding modes and calculated as follows,

$$\text{QS-score} = \frac{\sum_{\text{shared}(M,N)} w(\min(d_M, d_N))(1 - \frac{|d_M - d_N|}{12})}{\sum_{\text{shared}(M,N)} w(\min(d_M, d_N)) + \sum_{\text{non-shared}} w(d_M) + \sum_{\text{non-shared}} w(d_N)} \tag{8.2}$$

where M is the predicted multimeric model, and N is the corresponding experimental structure. d is the Euclidean distance between the Cβ (Cα for Glycin) atoms of shared or non-shared residue pairs within the protein multimer. Specifically, shared residue pairs are defined as those with a Cβ (Cα for Glycin) distance within 12Å, while non-shared residue pairs are those having a distance more than 12Å.

Overall, introducing these new accuracy measures will offer a deeper understanding of the generalizability of EquiRank, along with other existing methods for estimating the quality

of protein multimeric complexes.

Finally, we aim to leverage our interface quality estimation method to guide the refinement process for improving the structural accuracy of predicted multimeric protein complexes. Refinement of the predicted multimeric complexes helps in further improving their structural accuracy. While, the monomeric protein quality estimation has been an integral part and successfully integrated during the refinement of the monomeric protein model [20, 23], the application of multimeric quality estimation in the problem of refinement has yet to explore. Unlike most of the existing protein complex quality estimation methods, EquiRank estimates the error of individual interacting residue pairs within a protein multimer. It allows EquiRank not only to perform model ranking during the large-scale refinement sampling using the estimated global score but also to guide the refinement by employing its individual score of interacting residue pairs as constraints.

Bibliography

[1] F. Boadu, H. Cao, and J. Cheng, "Combining protein sequences and structures with transformers and equivariant graph neural networks to predict protein function," *Bioinformatics*, vol. 39, no. Supplement_1, pp. i318–i325, Jun. 2023. [Online]. Available: https://doi.org/10.1093/bioinformatics/btad208

[2] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Å½Ãdek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, no. 7873, pp. 583–589, Aug. 2021, number: 7873 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41586-021-03819-2

[3] J. O. Nealon, L. S. Philomina, and L. J. McGuffin, "Predictive and Experimental Approaches for Elucidating Protein–Protein Interactions and Quaternary Structures," *International Journal of Molecular Sciences*, vol. 18, no. 12, p. 2623, Dec. 2017. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5751226/

[4] N. V. Dokholyan, "Experimentally-Driven Protein Structure Modeling," *Journal of proteomics*, vol. 220, p. 103777, May 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7214187/

[5] K. Tunyasuvunakool, J. Adler, Z. Wu, T. Green, M. Zielinski, A. Žídek, A. Bridgland, A. Cowie, C. Meyer, A. Laydon, S. Velankar, G. J. Kleywegt, A. Bateman, R. Evans, A. Pritzel, M. Figurnov, O. Ronneberger, R. Bates, S. A. A. Kohl, A. Potapenko, A. J.

Ballard, B. Romera-Paredes, S. Nikolov, R. Jain, E. Clancy, D. Reiman, S. Petersen, A. W. Senior, K. Kavukcuoglu, E. Birney, P. Kohli, J. Jumper, and D. Hassabis, "Highly accurate protein structure prediction for the human proteome," *Nature*, vol. 596, no. 7873, pp. 590–596, Aug. 2021, number: 7873 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41586-021-03828-1

[6] R. Pearce and Y. Zhang, "Toward the solution of the protein structure prediction problem," *The Journal of Biological Chemistry*, vol. 297, no. 1, p. 100870, Jun. 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8254035/

[7] L. Heo and M. Feig, "Experimental accuracy in protein structure refinement via molecular dynamics simulations," *Proceedings of the National Academy of Sciences*, vol. 115, no. 52, pp. 13 276–13 281, Dec. 2018, publisher: National Academy of Sciences Section: Biological Sciences. [Online]. Available: https://www.pnas.org/content/115/52/13276

[8] B. Kuhlman and P. Bradley, "Advances in protein structure prediction and design," *Nature Reviews Molecular Cell Biology*, vol. 20, no. 11, pp. 681–697, Nov. 2019, number: 11 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41580-019-0163-x

[9] M. Baek, F. DiMaio, I. Anishchenko, J. Dauparas, S. Ovchinnikov, G. R. Lee, J. Wang, Q. Cong, L. N. Kinch, R. D. Schaeffer, C. MillÃ¡n, H. Park, C. Adams, C. R. Glassman, A. DeGiovanni, J. H. Pereira, A. V. Rodrigues, A. A. van Dijk, A. C. Ebrecht, D. J. Opperman, T. Sagmeister, C. Buhlheller, T. Pavkov-Keller, M. K. Rathinaswamy, U. Dalwadi, C. K. Yip, J. E. Burke, K. C. Garcia, N. V. Grishin, P. D. Adams, R. J. Read, and D. Baker, "Accurate prediction of protein structures and interactions using a three-track neural network," *Science*, vol. 373, no. 6557, pp. 871–876, Aug. 2021, publisher: American Association for the Advancement of Science. [Online]. Available: https://www.science.org/doi/10.1126/science.abj8754

[10] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, A. dos Santos Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, and A. Rives, "Evolutionary-scale prediction of atomic-level protein structure with a language model," *Science*, vol. 379, no. 6637, pp. 1123–1130, Mar. 2023, publisher: American Association for the Advancement of Science. [Online]. Available: https://www.science.org/doi/10.1126/science.ade2574

[11] X. Fang, F. Wang, L. Liu, J. He, D. Lin, Y. Xiang, K. Zhu, X. Zhang, H. Wu, H. Li, and L. Song, "A method for multiple-sequence-alignment-free protein structure prediction using a protein language model," *Nature Machine Intelligence*, vol. 5, no. 10, pp. 1087–1096, Oct. 2023, number: 10 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s42256-023-00721-6

[12] X. Jing and J. Xu, "Improved Protein Model Quality Assessment By Integrating Sequential And Pairwise Features Using Deep Learning," *Bioinformatics (Oxford, England)*, Dec. 2020.

[13] J. Won, M. Baek, B. Monastyrskyy, A. Kryshtafovych, and C. Seok, "Assessment of Protein Model Structure Accuracy Estimation in CASP13: Challenges in the Era of Deep Learning," *Proteins*, vol. 87, no. 12, pp. 1351–1360, Dec. 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6851486/

[14] K. Uziela, D. MenÃ©ndez Hurtado, N. Shu, B. Wallner, and A. Elofsson, "ProQ3D: improved model quality assessments using deep learning," *Bioinformatics*, vol. 33, no. 10, pp. 1578–1580, May 2017, publisher: Oxford Academic. [Online]. Available: https://academic.oup.com/bioinformatics/article/33/10/1578/2801464

[15] R. Cao, D. Bhattacharya, J. Hou, and J. Cheng, "DeepQA: improving the estimation of single protein model quality with deep belief networks,"

BMC Bioinformatics, vol. 17, p. 495, Dec. 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5139030/

[16] M. Kalman and N. Ben-Tal, "Quality assessment of protein model-structures using evolutionary conservation," *Bioinformatics*, vol. 26, no. 10, pp. 1299–1307, May 2010. [Online]. Available: https://academic.oup.com/bioinformatics/article/26/10/1299/193004

[17] A. F. Moumbock, J. Li, P. Mishra, M. Gao, and S. GÃ¼nther, "Current computational methods for predicting protein interactions of natural products," *Computational and Structural Biotechnology Journal*, vol. 17, pp. 1367–1376, Oct. 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6861622/

[18] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, A. d. S. Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, and A. Rives, "Evolutionary-scale prediction of atomic level protein structure with a language model," Oct. 2022, pages: 2022.07.20.500902 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2022.07.20.500902v2

[19] D. T. Jones and J. M. Thornton, "The impact of AlphaFold2 one year on," *Nature Methods*, vol. 19, no. 1, pp. 15–20, Jan. 2022, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41592-021-01365-3

[20] N. Hiranuma, H. Park, I. Anishchanka, M. Baek, and D. Baker, "Improved protein structure refinement guided by deep learning based accuracy estimation," Bioinformatics, preprint, Jul. 2020. [Online]. Available: http://biorxiv.org/lookup/doi/10.1101/2020.07.17.209643

[21] M. H. Shuvo, S. Bhattacharya, and D. Bhattacharya, "QDeep: distance-based protein model quality estimation by residue-level ensemble error classifications using stacked

deep residual neural networks," *Bioinformatics*, vol. 36, no. Supplement_1, pp. i285–i291, Jul. 2020. [Online]. Available: https://doi.org/10.1093/bioinformatics/btaa455

[22] D. Bhattacharya and M. H. Shuvo, "refineD: Protein structure refinement using machine learning guided restrained relaxation," *CASP13 abstract*, p. 32, 2018.

[23] M. H. Shuvo, M. Gulfam, and D. Bhattacharya, "DeepRefiner: high-accuracy protein structure refinement by deep network calibration," *Nucleic Acids Research*, vol. 49, no. W1, pp. W147–W152, Jul. 2021. [Online]. Available: https://doi.org/10.1093/nar/gkab361

[24] R. Alapati and D. Bhattacharya, "clustQ: Efficient Protein Decoy Clustering Using Superposition-free Weighted Internal Distance Comparisons," in *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, ser. BCB '18. New York, NY, USA: Association for Computing Machinery, Aug. 2018, pp. 307–314. [Online]. Available: https://doi.org/10.1145/3233547.3233570

[25] P. Benkert, S. C. E. Tosatto, and T. Schwede, "Global and local model quality estimation at CASP8 using the scoring functions QMEAN and QMEANclust," *Proteins*, vol. 77 Suppl 9, pp. 173–180, 2009.

[26] J. Cheng, Z. Wang, A. N. Tegge, and J. Eickholt, "Prediction of global and local quality of CASP8 models by MULTICOM series," *Proteins: Structure, Function, and Bioinformatics*, vol. 77, no. S9, pp. 181–184, 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.22487

[27] M. H. Shuvo, M. Karim, R. Roche, and D. Bhattacharya, "PIQLE: protein-protein interface quality estimation by deep graph learning of multimeric interaction geometries," *bioRxiv*, 2023, publisher: Cold Spring Harbor Laboratory _eprint:

https://www.biorxiv.org/content/early/2023/02/15/2023.02.14.528528.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2023/02/15/2023.02.14.528528

[28] L. Guo, J. He, P. Lin, S.-Y. Huang, and J. Wang, "TRScore: a 3D RepVGG-based scoring method for ranking protein docking models," *Bioinformatics*, vol. 38, no. 9, pp. 2444–2451, May 2022. [Online]. Available: https://doi.org/10.1093/bioinformatics/btac120

[29] K. OlechnoviÄ and . Venclovas, "VoroIF-GNN: Voronoi tessellation-derived protein-protein interface assessment using a graph neural network," *Proteins*, Jul. 2023.

[30] G. PagÃ¨s, B. Charmettant, and S. Grudinin, "Protein model quality assessment using 3D oriented convolutional neural networks," *Bioinformatics*, vol. 35, no. 18, pp. 3313–3319, Sep. 2019, publisher: Oxford Academic. [Online]. Available: https://academic.oup.com/bioinformatics/article/35/18/3313/5341430

[31] R. Sato and T. Ishida, "Protein model accuracy estimation based on local structure quality assessment using 3D convolutional neural network," *PLOS ONE*, vol. 14, no. 9, p. e0221347, Sep. 2019, publisher: Public Library of Science. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0221347

[32] D. M. Hurtado, K. Uziela, and A. Elofsson, "Deep transfer learning in the assessment of the quality of protein models," *arXiv:1804.06281 [q-bio]*, Apr. 2018, arXiv: 1804.06281. [Online]. Available: http://arxiv.org/abs/1804.06281

[33] X. Wang, G. Terashi, C. W. Christoffer, M. Zhu, and D. Kihara, "Protein docking model evaluation by 3D deep convolutional neural networks," *Bioinformatics (Oxford, England)*, vol. 36, no. 7, pp. 2113–2118, Apr. 2020.

[34] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," Dec. 2015, arXiv:1511.08458 [cs]. [Online]. Available: http://arxiv.org/abs/1511.08458

[35] S.-S. Guo, J. Liu, X.-G. Zhou, and G.-J. Zhang, "DeepUMQA: ultrafast shape recognition-based protein model quality assessment using deep learning," *Bioinformatics*, vol. 38, no. 7, pp. 1895–1903, Apr. 2022. [Online]. Available: https://doi.org/10.1093/bioinformatics/btac056

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778, iSSN: 1063-6919.

[37] F. Baldassarre, D. Menéndez Hurtado, A. Elofsson, and H. Azizpour, "GraphQA: protein model quality assessment using graph convolutional networks," *Bioinformatics*, vol. 37, no. 3, pp. 360–366, Apr. 2021. [Online]. Available: https://doi.org/10.1093/bioinformatics/btaa714

[38] S. Sanyal, I. Anishchenko, A. Dagar, D. Baker, and P. Talukdar, "ProteinGCN: Protein model quality assessment using Graph Convolutional Networks," *bioRxiv*, p. 2020.04.06.028266, Apr. 2020, publisher: Cold Spring Harbor Laboratory Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2020.04.06.028266v1

[39] X. Wang, S. T. Flannery, and D. Kihara, "Protein Docking Model Evaluation by Graph Neural Networks," *Frontiers in Molecular Biosciences*, vol. 8, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmolb.2021.647915

[40] M. McFee and P. M. Kim, "GDockScore: a graph-based proteinâ€"protein docking scoring function," *Bioinformatics Advances*, vol. 3, no. 1, p. vbad072, Jan. 2023. [Online]. Available: https://doi.org/10.1093/bioadv/vbad072

[41] X. Xu and A. M. J. J. Bonvin, "DeepRank-GNN-esm: A Graph Neural Network for Scoring Protein-Protein Models using Protein Language Model,"

Jun. 2023, pages: 2023.06.22.546080 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2023.06.22.546080v1

[42] C. Chen, X. Chen, A. Morehead, T. Wu, and J. Cheng, "3D-equivariant graph neural networks for protein model quality assessment," *Bioinformatics*, vol. 39, no. 1, p. btad030, Jan. 2023. [Online]. Available: https://doi.org/10.1093/bioinformatics/btad030

[43] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *arXiv:1609.02907 [cs, stat]*, Feb. 2017, arXiv: 1609.02907. [Online]. Available: http://arxiv.org/abs/1609.02907

[44] P. VeliÄ koviÄ‡, G. Cucurull, A. Casanova, A. Romero, P. LiÃ², and Y. Bengio, "Graph Attention Networks," *arXiv:1710.10903 [cs, stat]*, Feb. 2018, arXiv: 1710.10903. [Online]. Available: http://arxiv.org/abs/1710.10903

[45] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph Transformer Networks," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/9d63484abb477c97640154d40595a3bb-Abstract.html

[46] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) Equivariant Graph Neural Networks," Feb. 2022, arXiv:2102.09844 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2102.09844

[47] Z. Wang, A. N. Tegge, and J. Cheng, "Evaluating the absolute quality of a single protein model using structural features and support vector machines," *Proteins*, vol. 75, no. 3, pp. 638–647, May 2009.

[48] B. Manavalan, J. Lee, and J. Lee, "Random Forest-Based Protein Model Quality Assessment (RFMQA) Using Structural Features and Potential Energy

Terms," *PLOS ONE*, vol. 9, no. 9, p. e106542, Sep. 2014. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0106542

[49] A. Ray, E. Lindahl, and B. Wallner, "Improved model quality assessment using ProQ2," *BMC Bioinformatics*, vol. 13, no. 1, p. 224, Sep. 2012. [Online]. Available: https://doi.org/10.1186/1471-2105-13-224

[50] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks," Aug. 2020, arXiv:1909.01315 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1909.01315

[51] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2666651021000012

[52] J. G. Greener, S. M. Kandathil, and D. T. Jones, "Deep learning extends de novo protein modelling coverage of genomes using iteratively predicted structural constraints," *Nature Communications*, vol. 10, no. 1, p. 3977, Sep. 2019, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41467-019-11994-0

[53] O.-E. Ganea, X. Huang, C. Bunne, Y. Bian, R. Barzilay, T. Jaakkola, and A. Krause, "Independent SE(3)-Equivariant Models for End-to-End Rigid Protein Docking," Mar. 2022, arXiv:2111.07786 [cs]. [Online]. Available: http://arxiv.org/abs/2111.07786

[54] M. H. Shuvo, M. Karim, and D. Bhattacharya, "iQDeep: an antegrated web server for protein scoring using multiscale deep learning models," *Journal of Molecular Biology*, p. 168057, Mar. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022283623001134

[55] K. Uziela, N. Shu, B. Wallner, and A. Elofsson, "ProQ3: Improved model quality assessments using Rosetta energy terms," *Scientific Reports*, vol. 6, no. 1, p. 33509, Oct. 2016, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/srep33509

[56] G. Derevyanko, S. Grudinin, Y. Bengio, and G. Lamoureux, "Deep convolutional networks for quality assessment of protein folds," *Bioinformatics*, vol. 34, no. 23, pp. 4046–4053, Dec. 2018, publisher: Oxford Academic. [Online]. Available: https://academic.oup.com/bioinformatics/article/34/23/4046/5040325

[57] M. Karasikov, G. PagÃ¨s, and S. Grudinin, "Smooth orientation-dependent scoring function for coarse-grained protein quality assessment," *Bioinformatics*, vol. 35, no. 16, pp. 2801–2808, Aug. 2019. [Online]. Available: https://academic.oup.com/bioinformatics/article/35/16/2801/5258107

[58] K. OlechnoviÄ , E. KulberkytÄ—, and . Venclovas, "CAD-score: A new contact area difference-based function for evaluation of protein structural models," *Proteins: Structure, Function, and Bioinformatics*, vol. 81, no. 1, pp. 149–162, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.24172

[59] J. Cheng, M.-H. Choe, A. Elofsson, K.-S. Han, J. Hou, A. H. A. Maghrabi, L. J. McGuffin, D. MenÃ©ndez-Hurtado, K. OlechnoviÄ , T. Schwede, G. Studer, K. Uziela, . Venclovas, and B. Wallner, "Estimation of model accuracy in CASP13," *Proteins*, vol. 87, no. 12, pp. 1361–1377, Dec. 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6851425/

[60] L. J. McGuffin and D. B. Roche, "Rapid model quality assessment for protein structure predictions using the comparison of multiple models without structural alignments," *Bioinformatics (Oxford, England)*, vol. 26, no. 2, pp. 182–188, Jan. 2010.

[61] B. Manavalan and J. Lee, "SVMQA: supportâ€"vector-machine-based protein

single-model quality assessment," *Bioinformatics*, vol. 33, no. 16, pp. 2496–2503, Aug. 2017. [Online]. Available: https://academic.oup.com/bioinformatics/article/33/16/2496/3611273

[62] A. Kryshtafovych, B. Monastyrskyy, K. Fidelis, T. Schwede, and A. Tramontano, "Assessment of model accuracy estimations in CASP12," *Proteins: Structure, Function, and Bioinformatics*, vol. 86, no. S1, pp. 345–360, 2018, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.25371. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.25371

[63] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Å½Ãdek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis, "Improved protein structure prediction using potentials from deep learning," *Nature*, vol. 577, no. 7792, pp. 706–710, Jan. 2020, number: 7792 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41586-019-1923-7

[64] J. Xu, "Distance-based protein folding powered by deep learning," *Proceedings of the National Academy of Sciences*, vol. 116, no. 34, pp. 16 856–16 865, Aug. 2019, publisher: National Academy of Sciences Section: PNAS Plus. [Online]. Available: https://www.pnas.org/content/116/34/16856

[65] Y. Li, J. Hu, C. Zhang, D.-J. Yu, and Y. Zhang, "ResPRE: high-accuracy protein contact prediction by coupling precision matrix with deep residual neural networks," *Bioinformatics*, vol. 35, no. 22, pp. 4647–4655, Nov. 2019. [Online]. Available: https://academic.oup.com/bioinformatics/article/35/22/4647/5487385

[66] S. Wang, S. Sun, Z. Li, R. Zhang, and J. Xu, "Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model," *PLOS*

Computational Biology, vol. 13, no. 1, p. e1005324, Jan. 2017. [Online]. Available: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005324

[67] A. Zemla, "LGA: A method for finding 3D similarities in protein structures," *Nucleic Acids Research*, vol. 31, no. 13, pp. 3370–3374, Jul. 2003.

[68] M. Remmert, A. Biegert, A. Hauser, and J. SÃ¶ding, "HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment," *Nature Methods*, vol. 9, no. 2, pp. 173–175, Feb. 2012. [Online]. Available: https://www.nature.com/articles/nmeth.1818

[69] M. Mirdita, L. von den Driesch, C. Galiez, M. J. Martin, J. SÃ¶ding, and M. Steinegger, "Uniclust databases of clustered and deeply annotated protein sequences and alignments," *Nucleic Acids Research*, vol. 45, no. D1, pp. D170–D176, Jan. 2017.

[70] B. E. Suzek, Y. Wang, H. Huang, P. B. McGarvey, and C. H. Wu, "UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches," *Bioinformatics*, vol. 31, no. 6, pp. 926–932, Mar. 2015. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4375400/

[71] M. Steinegger and J. SÃ¶ding, "Clustering huge protein sequence sets in linear time," *Nature Communications*, vol. 9, no. 1, p. 2542, 2018.

[72] C. Zhang, W. Zheng, S. M. Mortuza, Y. Li, and Y. Zhang, "DeepMSA: constructing deep multiple sequence alignment to improve contact prediction and fold-recognition for distant-homology proteins," *Bioinformatics*, vol. 36, no. 7, pp. 2105–2112, Apr. 2020, publisher: Oxford Academic. [Online]. Available: https://academic.oup.com/bioinformatics/article/36/7/2105/5628221

[73] P. Di Lena, P. Fariselli, L. Margara, M. Vassura, and R. Casadio, "Fast overlapping of protein contact maps by alignment of eigenvectors," *Bioinformatics*,

vol. 26, no. 18, pp. 2250–2258, Sep. 2010. [Online]. Available: https://academic.oup.com/bioinformatics/article/26/18/2250/207560

[74] S. F. Altschul, T. L. Madden, A. A. SchÃ¤ffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs." *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, Sep. 1997. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC146917/

[75] R. Heffernan, Y. Yang, K. Paliwal, and Y. Zhou, "Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility," *Bioinformatics (Oxford, England)*, vol. 33, no. 18, pp. 2842–2849, Sep. 2017.

[76] A. Leaver-Fay, M. Tyka, S. M. Lewis, O. F. Lange, J. Thompson, R. Jacak, K. Kaufman, P. D. Renfrew, C. A. Smith, W. Sheffler, I. W. Davis, S. Cooper, A. Treuille, D. J. Mandell, F. Richter, Y.-E. A. Ban, S. J. Fleishman, J. E. Corn, D. E. Kim, S. Lyskov, M. Berrondo, S. Mentzer, Z. PopoviÄ‡, J. J. Havranek, J. Karanicolas, R. Das, J. Meiler, T. Kortemme, J. J. Gray, B. Kuhlman, D. Baker, and P. Bradley, "ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules," *Methods in Enzymology*, vol. 487, pp. 545–574, 2011.

[77] C. A. Rohl, C. E. M. Strauss, K. M. S. Misura, and D. Baker, "Protein structure prediction using Rosetta," *Methods in Enzymology*, vol. 383, pp. 66–93, 2004.

[78] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference on Machine Learning*, Jun. 2015, pp. 448–456. [Online]. Available: http://proceedings.mlr.press/v37/ioffe15.html

[79] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, and A. Tramontano, "Critical

assessment of methods of protein structure prediction: Progress and new directions in round XI," *Proteins*, vol. 84 Suppl 1, pp. 4–14, 2016.

[80] ——, "Critical assessment of methods of protein structure prediction (CASP)-Round XII," *Proteins*, vol. 86 Suppl 1, pp. 7–15, Mar. 2018.

[81] J. Zhang and D. Xu, "Fast algorithm for population-based protein structural model analysis," *Proteomics*, vol. 13, no. 2, pp. 221–229, Jan. 2013.

[82] F. Chollet, "Keras: Deep learning library for theano and tensorflow," 2015. [Online]. Available: https://keras.io/

[83] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Dec. 2014. [Online]. Available: https://arxiv.org/abs/1412.6980v9

[84] A. Kryshtafovych, A. Barbato, B. Monastyrskyy, K. Fidelis, T. Schwede, and A. Tramontano, "Methods of model accuracy estimation can help selecting the best models from decoy sets: Assessment of model accuracy estimations in CASP11," *Proteins*, vol. 84 Suppl 1, pp. 349–369, 2016.

[85] N. Kalisman, A. Levi, T. Maximova, D. Reshef, S. Zafriri-Lynn, Y. Gleyzer, and C. Keasar, "MESHI: a new library of Java classes for molecular modeling," *Bioinformatics*, vol. 21, no. 20, pp. 3931–3932, Oct. 2005, publisher: Oxford Academic. [Online]. Available: https://academic.oup.com/bioinformatics/article/21/20/3931/202988

[86] K. OlechnoviÄ and . Venclovas, "VoroMQA: Assessment of protein structure quality using interatomic contact areas," *Proteins: Structure, Function, and Bioinformatics*, vol. 85, no. 6, pp. 1131–1145, 2017, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.25278. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.25278

[87] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[88] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: Association for Computing Machinery, Jun. 2009, pp. 609–616. [Online]. Available: https://doi.org/10.1145/1553374.1553453

[89] B. Wallner and A. Elofsson, "Can correct protein models be identified?" *Protein Science*, vol. 12, no. 5, pp. 1073–1086, 2003, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1110/ps.0236803. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1110/ps.0236803

[90] S. Kwon, J. Won, A. Kryshtafovych, and C. Seok, "Assessment of protein model structure accuracy estimation in CASP14: Old and new challenges," *Proteins*, vol. 89, no. 12, pp. 1940–1948, Dec. 2021.

[91] A. Kryshtafovych, A. Barbato, K. Fidelis, B. Monastyrskyy, T. Schwede, and A. Tramontano, "Assessment of the assessment: evaluation of the model quality estimates in CASP10," *Proteins*, vol. 82 Suppl 2, pp. 112–126, Feb. 2014.

[92] A. Elofsson, K. Joo, C. Keasar, J. Lee, A. H. A. Maghrabi, B. Manavalan, L. J. McGuffin, D. M. Hurtado, C. Mirabello, R. PilstÃ¥l, T. Sidi, K. Uziela, and B. Wallner, "Methods for estimation of model accuracy in CASP12," *Proteins: Structure, Function, and Bioinformatics*, vol. 86, no. S1, pp. 361–373, 2018, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.25395. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.25395

[93] G. Ahdritz, N. Bouatta, S. Kadyan, Q. Xia, W. Gerecke, T. J. Oâ€™Donnell,

D. Berenberg, I. Fisk, N. Zanichelli, B. Zhang, A. Nowaczynski, B. Wang, M. M. Stepniewska-Dziubinska, S. Zhang, A. Ojewole, M. E. Guney, S. Biderman, A. M. Watkins, S. Ra, P. R. Lorenzo, L. Nivon, B. Weitzner, Y.-E. A. Ban, P. K. Sorger, E. Mostaque, Z. Zhang, R. Bonneau, and M. AlQuraishi, "OpenFold: Retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization," Nov. 2022, pages: 2022.11.20.517210 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2022.11.20.517210v2

[94] X. Jing and J. Xu, "Fast and effective protein model refinement using deep graph neural networks," *Nature computational science*, vol. 1, no. 7, pp. 462–469, Jul. 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8939834/

[95] L. Heo and M. Feig, "High-accuracy protein structures by combining machine-learning with physics-based refinement," *Proteins: Structure, Function, and Bioinformatics*, vol. 88, no. 5, pp. 637–642, 2020, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.25847. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.25847

[96] D. Bhattacharya, "refineD: improved protein structure refinement using machine learning based restrained relaxation," *Bioinformatics*, vol. 35, no. 18, pp. 3320–3328, Sep. 2019, publisher: Oxford Academic. [Online]. Available: https://academic.oup.com/bioinformatics/article/35/18/3320/5317159

[97] J. Kopp, L. Bordoli, J. N. D. Battey, F. Kiefer, and T. Schwede, "Assessment of CASP7 predictions for template-based modeling targets," *Proteins*, vol. 69 Suppl 8, pp. 38–56, 2007.

[98] J. L. MacCallum, A. PÃ©rez, M. J. Schnieders, L. Hua, M. P. Jacobson, and K. A. Dill, "Assessment of protein structure refinement in CASP9," *Proteins*, vol. 79 Suppl 10, pp. 74–90, 2011.

[99] A. Kryshtafovych, K. Fidelis, and A. Tramontano, "Evaluation of model quality predictions in CASP9," *Proteins*, vol. 79 Suppl 10, pp. 91–106, 2011.

[100] N. Rego and D. Koes, "3Dmol.js: molecular visualization with WebGL," *Bioinformatics (Oxford, England)*, vol. 31, no. 8, pp. 1322–1324, Apr. 2015.

[101] J. Moult, K. Fidelis, A. Kryshtafovych, and A. Tramontano, "Critical Assessment of Methods of Protein Structure Prediction (CASP) - Round IX," *Proteins*, vol. 79, no. 0 10, pp. 1–5, 2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4180088/

[102] J. Moult, K. Fidelis, A. Kryshtafovych, T. Schwede, and A. Tramontano, "Critical assessment of methods of protein structure prediction (CASP) â€" round x," *Proteins*, vol. 82, no. 0 2, pp. 1–6, Feb. 2014. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4394854/

[103] R. Heffernan, K. Paliwal, J. Lyons, A. Dehzangi, A. Sharma, J. Wang, A. Sattar, Y. Yang, and Y. Zhou, "Improving prediction of secondary structure, local backbone angles, and solvent accessible surface area of proteins by iterative deep learning," *Scientific Reports*, vol. 5, p. 11476, Jun. 2015. [Online]. Available: https://www.nature.com/articles/srep11476

[104] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers*, vol. 22, no. 12, pp. 2577–2637, Dec. 1983.

[105] Z. Qin, F. Yu, C. Liu, and X. Chen, "How convolutional neural networks see the world — A survey of convolutional neural network visualization methods," *Mathematical Foundations of Computing*, vol. 1, no. 2, pp. 149–180, May 2018, publisher: Mathematical Foundations of Computing. [Online]. Available: https://www.aimsciences.org/en/article/doi/10.3934/mfc.2018008

[106] A. Y. Ng, "Feature selection, L_1 vs. L_2 regularization, and rotational invariance," in *Twenty-first international conference on Machine learning - ICML '04*. Banff, Alberta, Canada: ACM Press, 2004, p. 78. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1015330.1015435

[107] D. Bhattacharya, J. Nowotny, R. Cao, and J. Cheng, "3Drefine: an interactive web server for efficient protein structure refinement," *Nucleic Acids Research*, vol. 44, no. W1, pp. W406–409, Jul. 2016.

[108] H. Park, S. Ovchinnikov, D. E. Kim, F. DiMaio, and D. Baker, "Protein homology model refinement by large-scale energy optimization," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 115, no. 12, pp. 3054–3059, Mar. 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5866580/

[109] J. Yang, I. Anishchenko, H. Park, Z. Peng, S. Ovchinnikov, and D. Baker, "Improved protein structure prediction using predicted interresidue orientations," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 117, no. 3, pp. 1496–1503, Jan. 2020.

[110] A. Kryshtafovych, T. Schwede, M. Topf, K. Fidelis, and J. Moult, "Critical assessment of methods of protein structure prediction (CASP)-Round XIII," *Proteins*, vol. 87, no. 12, pp. 1011–1020, Dec. 2019.

[111] L. Heo, C. F. Arbour, and M. Feig, "Driven to near-experimental accuracy by refinement via molecular dynamics simulations," *Proteins: Structure, Function, and Bioinformatics*, vol. 87, no. 12, pp. 1263–1275, 2019, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.25759. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.25759

[112] H. Park, G. R. Lee, D. E. Kim, I. Anishchenko, Q. Cong, and D. Baker,

"High-accuracy refinement using Rosetta in CASP13," *Proteins: Structure, Function, and Bioinformatics*, vol. 87, no. 12, pp. 1276–1282, 2019, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.25784. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.25784

[113] N. Hiranuma, H. Park, M. Baek, I. Anishchenko, J. Dauparas, and D. Baker, "Improved protein structure refinement guided by deep learning based accuracy estimation," *Nature Communications*, vol. 12, no. 1, p. 1340, Feb. 2021.

[114] X. Cao and P. Tian, "Molecular free energy optimization on a computational graph," *RSC Advances*, vol. 11, no. 21, pp. 12 929–12 937, Mar. 2021, publisher: The Royal Society of Chemistry. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2021/ra/d1ra01455b

[115] V. B. Chen, W. B. Arendall, J. J. Headd, D. A. Keedy, R. M. Immormino, G. J. Kapral, L. W. Murray, J. S. Richardson, and D. C. Richardson, "MolProbity: all-atom structure validation for macromolecular crystallography," *Acta Crystallographica Section D: Biological Crystallography*, vol. 66, no. Pt 1, pp. 12–21, Jan. 2010. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2803126/

[116] H. Zhou and J. Skolnick, "GOAP: a generalized orientation-dependent, all-atom statistical potential for protein structure prediction," *Biophysical Journal*, vol. 101, no. 8, pp. 2043–2052, Oct. 2011.

[117] M. Lu, A. D. Dousis, and J. Ma, "OPUS-PSP: an orientation-dependent statistical all-atom potential derived from side-chain packing," *Journal of Molecular Biology*, vol. 376, no. 1, pp. 288–301, Feb. 2008.

[118] H. Zhou and Y. Zhou, "Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability pre-

diction," *Protein Science: A Publication of the Protein Society*, vol. 11, no. 11, pp. 2714–2726, Nov. 2002.

[119] J. Zhang and Y. Zhang, "A novel side-chain orientation dependent potential derived from random-walk reference state for protein fold selection and structure prediction," *PloS One*, vol. 5, no. 10, p. e15386, Oct. 2010.

[120] R. J. Read, M. D. Sammito, A. Kryshtafovych, and T. I. Croll, "Evaluation of model refinement in CASP13," *Proteins*, vol. 87, no. 12, pp. 1249–1262, Dec. 2019.

[121] S. Wang, J. Peng, J. Ma, and J. Xu, "Protein Secondary Structure Prediction Using Deep Convolutional Neural Fields," *Scientific Reports*, vol. 6, p. 18962, Jan. 2016.

[122] S. Wang, S. Sun, and J. Xu, "AUC-Maximized Deep Convolutional Neural Fields for Protein Sequence Labeling," *Machine learning and knowledge discovery in databases: European Conference, ECML PKDD ...: proceedings. ECML PKDD (Conference)*, vol. 9852, pp. 1–16, Sep. 2016.

[123] R. F. Alford, A. Leaver-Fay, J. R. Jeliazkov, M. J. Oâ€™Meara, F. P. DiMaio, H. Park, M. V. Shapovalov, P. D. Renfrew, V. K. Mulligan, K. Kappel, J. W. Labonte, M. S. Pacella, R. Bonneau, P. Bradley, R. L. Dunbrack, R. Das, D. Baker, B. Kuhlman, T. Kortemme, and J. J. Gray, "The Rosetta all-atom energy function for macromolecular modeling and design," *Journal of chemical theory and computation*, vol. 13, no. 6, pp. 3031–3048, Jun. 2017. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5717763/

[124] J. L. MacCallum, L. Hua, M. J. Schnieders, V. S. Pande, M. P. Jacobson, and K. A. Dill, "Assessment of the protein-structure refinement category in CASP8," *Proteins: Structure, Function, and Bioinformatics*, vol. 77, no. S9, pp. 66–80, 2009, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.22538. [Online].

Available: https://pericles.pericles-prod.literatumonline.com/doi/abs/10.1002/prot.22538

[125] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A*, vol. 32, no. 5, pp. 922–923, 1976, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1107/S0567739476001873. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1107/S0567739476001873

[126] T. Nugent, D. Cozzetto, and D. T. Jones, "Evaluation of predictions in the CASP10 model refinement category," *Proteins*, vol. 82 Suppl 2, pp. 98–111, Feb. 2014.

[127] L. Heo, H. Park, and C. Seok, "GalaxyRefine: Protein structure refinement driven by side-chain repacking," *Nucleic Acids Research*, vol. 41, no. Web Server issue, pp. W384–388, Jul. 2013.

[128] G. R. Lee, J. Won, L. Heo, and C. Seok, "GalaxyRefine2: simultaneous refinement of inaccurate local regions and overall protein structure," *Nucleic Acids Research*, vol. 47, no. W1, pp. W451–W455, Jul. 2019. [Online]. Available: https://doi.org/10.1093/nar/gkz288

[129] D. Xu and Y. Zhang, "Improving the physical realism and structural accuracy of protein models by a two-step atomic-level energy minimization," *Biophysical Journal*, vol. 101, no. 10, pp. 2525–2534, Nov. 2011.

[130] M. Heinig and D. Frishman, "STRIDE: a web server for secondary structure assignment from known atomic coordinates of proteins," *Nucleic Acids Research*, vol. 32, no. Web Server issue, pp. W500–W502, Jul. 2004. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC441567/

[131] F. Khatib, S. Cooper, M. D. Tyka, K. Xu, I. Makedon, Z. PopoviÄ‡, D. Baker, and F. Players, "Algorithm discovery by protein folding game

players," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, no. 47, pp. 18 949–18 953, Nov. 2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3223433/

[132] M. D. Tyka, D. A. Keedy, I. AndrÃ©, F. DiMaio, Y. Song, D. C. Richardson, J. S. Richardsonb, and D. Baker, "Alternate states of proteins revealed by detailed energy landscape mapping," *Journal of molecular biology*, vol. 405, no. 2, pp. 607–618, Jan. 2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3046547/

[133] L. Hovan, V. Oleinikovas, H. Yalinca, A. Kryshtafovych, G. Saladino, and F. L. Gervasio, "Assessment of the model refinement category in CASP12," *Proteins*, vol. 86 Suppl 1, pp. 152–167, Mar. 2018.

[134] X. Peng, J. Wang, W. Peng, F.-X. Wu, and Y. Pan, "Protein-protein interactions: detection, reliability assessment and applications," *Briefings in Bioinformatics*, vol. 18, no. 5, pp. 798–819, Sep. 2017.

[135] B. Wallner, "AFsample: Improving Multimer Prediction with AlphaFold using Aggressive Sampling," Dec. 2022, pages: 2022.12.20.521205 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2022.12.20.521205v2

[136] J. Zahiri, A. Emamjomeh, S. Bagheri, A. Ivazeh, G. Mahdevar, H. Sepasi Tehrani, M. Mirzaie, B. A. Fakheri, and M. Mohammad-Noori, "Protein complex prediction: A survey," *Genomics*, vol. 112, no. 1, pp. 174–183, Jan. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S088875431830572X

[137] P. Bryant, G. Pozzati, and A. Elofsson, "Improved prediction of protein-protein interactions using AlphaFold2," *Nature Communications*, vol. 13, no. 1, p. 1265, Mar. 2022, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41467-022-28865-w

[138] R. Evans, M. Oâ€™Neill, A. Pritzel, N. Antropova, A. Senior, T. Green, A. Å½Ãdek, R. Bates, S. Blackwell, J. Yim, O. Ronneberger, S. Bodenstein, M. Zielinski, A. Bridgland, A. Potapenko, A. Cowie, K. Tunyasuvunakool, R. Jain, E. Clancy, P. Kohli, J. Jumper, and D. Hassabis, "Protein complex prediction with AlphaFold-Multimer," Mar. 2022, pages: 2021.10.04.463034 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2021.10.04.463034v2

[139] C. Christoffer, V. Bharadwaj, R. Luu, and D. Kihara, "LZerD Protein-Protein Docking Webserver Enhanced With de novo Structure Prediction," *Frontiers in Molecular Biosciences*, vol. 8, 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fmolb.2021.724947

[140] B. G. Pierce, K. Wiehe, H. Hwang, B.-H. Kim, T. Vreven, and Z. Weng, "ZDOCK server: interactive docking prediction of proteinâ€“protein complexes and symmetric multimers," *Bioinformatics*, vol. 30, no. 12, pp. 1771–1773, Jun. 2014. [Online]. Available: https://doi.org/10.1093/bioinformatics/btu097

[141] S. Lyskov and J. J. Gray, "The RosettaDock server for local protein-protein docking," *Nucleic Acids Research*, vol. 36, no. Web Server issue, pp. W233–238, Jul. 2008.

[142] M. Gao, D. Nakajima An, J. M. Parks, and J. Skolnick, "AF2Complex predicts direct physical interactions in multimeric proteins with deep learning," *Nature Communications*, vol. 13, no. 1, p. 1744, Apr. 2022, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41467-022-29394-2

[143] V. Sandor and D. Kozakov, "Sampling and scoring: A marriage made in heaven," *Proteins*, vol. 81, no. 11, pp. 1874–1884, Nov. 2013. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3942495/

[144] Y. Cao and Y. Shen, "Energy-based Graph Convolutional Networks for Scoring

Protein Docking Models," *Proteins*, vol. 88, no. 8, pp. 1091–1099, Aug. 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7374013/

[145] B. Pierce and Z. Weng, "ZRANK: reranking protein docking predictions with an optimized energy function," *Proteins*, vol. 67, no. 4, pp. 1078–1086, Jun. 2007.

[146] ——, "A combination of rescoring and refinement significantly improves protein docking performance," *Proteins*, vol. 72, no. 1, pp. 270–279, Jul. 2008.

[147] S.-Y. Huang and X. Zou, "An iterative knowledge-based scoring function for protein-protein recognition," *Proteins*, vol. 72, no. 2, pp. 557–579, Aug. 2008.

[148] X. Chen, A. Morehead, J. Liu, and J. Cheng, "DProQ: A Gated-Graph Transformer for Protein Complex Structure Assessment," May 2022, pages: 2022.05.19.492741 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2022.05.19.492741v2

[149] B. Dai and C. Bailey-Kellogg, "Protein interaction interface region prediction by geometric deep learning," *Bioinformatics*, vol. 37, no. 17, pp. 2580–2588, Sep. 2021. [Online]. Available: https://doi.org/10.1093/bioinformatics/btab154

[150] N. A. Marze, S. S. Roy Burman, W. Sheffler, and J. J. Gray, "Efficient flexible backbone proteinâ€"protein docking for challenging targets," *Bioinformatics*, vol. 34, no. 20, pp. 3461–3469, Oct. 2018. [Online]. Available: https://doi.org/10.1093/bioinformatics/bty355

[151] C. Zhu, Y. Gao, H. Li, S. Meng, L. Li, J. S. Francisco, and X. C. Zeng, "Characterizing hydrophobicity of amino acid side chains in a protein environment via measuring contact angle of a water nanodroplet on planar peptide network," *Proceedings of the National Academy of Sciences*, vol. 113, no. 46, pp. 12 946–12 951, Nov. 2016, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: https://www.pnas.org/doi/10.1073/pnas.1616138113

[152] K. Kumar, S. M. Woo, T. Siu, W. A. Cortopassi, F. Duarte, and R. S. Paton, "Cationâ€"Ï€ interactions in proteinâ€"ligand binding: theory and data-mining reveal different roles for lysine and arginine," *Chemical Science*, vol. 9, no. 10, pp. 2655–2665, Mar. 2018, publisher: The Royal Society of Chemistry. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2018/sc/c7sc04905f

[153] M. Tavafoghi and M. Cerruti, "The role of amino acids in hydroxyapatite mineralization," *Journal of The Royal Society Interface*, vol. 13, no. 123, p. 20160462, Oct. 2016, publisher: Royal Society. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rsif.2016.0462

[154] H. Li, J. Hou, B. Adhikari, Q. Lyu, and J. Cheng, "Deep learning methods for protein torsion angle prediction," *BMC Bioinformatics*, vol. 18, no. 1, p. 417, Sep. 2017. [Online]. Available: https://doi.org/10.1186/s12859-017-1834-2

[155] M. Remmert, A. Biegert, A. Hauser, and J. SÃ¶ding, "HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment," *Nature Methods*, vol. 9, no. 2, pp. 173–175, Dec. 2011.

[156] M. H. Shuvo, S. Bhattacharya, R. Roche, and D. Bhattacharya, "Protein tertiary structure prediction by Bhattacharya group in CASP14," *CASP14 abstract*, p. 38, 2020.

[157] Z. Xie and J. Xu, "Deep graph learning of inter-protein contacts," *Bioinformatics*, vol. 38, no. 4, pp. 947–953, Feb. 2022. [Online]. Available: https://doi.org/10.1093/bioinformatics/btab761

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

[159] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking Graph Neural Networks," May 2022, arXiv:2003.00982 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2003.00982

[160] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, Mar. 2010, pp. 249–256, iSSN: 1938-7228. [Online]. Available: https://proceedings.mlr.press/v9/glorot10a.html

[161] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks," Aug. 2020, arXiv:1909.01315 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1909.01315

[162] P. J. Kundrotas, I. Anishchenko, T. Dauzhenka, I. Kotthoff, D. Mnevets, M. M. Copeland, and I. A. Vakser, "Dockground: A comprehensive data resource for modeling of protein complexes," *Protein Science: A Publication of the Protein Society*, vol. 27, no. 1, pp. 172–181, Jan. 2018.

[163] X. Chen, A. Morehead, J. Liu, and J. Cheng, "A Gated Graph Transformer for Protein Complex Structure Quality Assessment and its Performance in CASP15," Feb. 2023, pages: 2022.05.19.492741 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2022.05.19.492741v3

[164] H. Hwang, T. Vreven, J. Janin, and Z. Weng, "Protein-protein docking benchmark version 4.0," *Proteins*, vol. 78, no. 15, pp. 3111–3114, Nov. 2010.

[165] S. Basu and B. Wallner, "DockQ: A Quality Measure for Protein-Protein Docking Models," *PLOS ONE*, vol. 11, no. 8, p. e0161879, Aug. 2016, publisher: Public

Library of Science. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0161879

[166] M. F. Lensink and S. J. Wodak, "Docking, scoring, and affinity prediction in CAPRI," *Proteins: Structure, Function, and Bioinformatics*, vol. 81, no. 12, pp. 2082–2095, 2013, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.24428. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.24428

[167] J. P. Roney and S. Ovchinnikov, "State-of-the-Art Estimation of Protein Model Accuracy Using AlphaFold," *Physical Review Letters*, vol. 129, no. 23, p. 238101, Nov. 2022, publisher: American Physical Society. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.129.238101

[168] H. Lu, Q. Zhou, J. He, Z. Jiang, C. Peng, R. Tong, and J. Shi, "Recent advances in the development of proteinâ€"protein interactions modulators: mechanisms and clinical trials," *Signal Transduction and Targeted Therapy*, vol. 5, no. 1, pp. 1–23, Sep. 2020, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41392-020-00315-3

[169] N. Zaki, D. Efimov, and J. Berengueres, "Protein complex detection using interaction reliability assessment and weighted clustering coefficient," *BMC Bioinformatics*, vol. 14, no. 1, p. 163, May 2013. [Online]. Available: https://doi.org/10.1186/1471-2105-14-163

[170] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs]*, Dec. 2015, arXiv: 1512.03385. [Online]. Available: http://arxiv.org/abs/1512.03385

[171] A. Keramatfar, M. Rafiee, and H. Amirkhani, "Graph Neural Networks: A bibliometrics overview," *Machine Learning with Applications*, vol. 10, p. 100401,

Dec. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666827022000780

[172] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, D. Bhowmik, and B. Rost, "ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 10, pp. 7112–7127, Oct. 2022.

[173] N. Ferruz, S. Schmidt, and B. HÃ¶cker, "ProtGPT2 is a deep unsupervised language model for protein design," *Nature Communications*, vol. 13, no. 1, p. 4348, Jul. 2022, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41467-022-32007-7

[174] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C. L. Zitnick, J. Ma, and R. Fergus, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *Proceedings of the National Academy of Sciences*, vol. 118, no. 15, p. e2016239118, Apr. 2021, publisher: Proceedings of the National Academy of Sciences. [Online]. Available: https://www.pnas.org/doi/10.1073/pnas.2016239118

[175] R. Chowdhury, N. Bouatta, S. Biswas, C. Floristean, A. Kharkar, K. Roy, C. Rochereau, G. Ahdritz, J. Zhang, G. M. Church, P. K. Sorger, and M. AlQuraishi, "Single-sequence protein structure prediction using a language model and deep learning," *Nature Biotechnology*, vol. 40, no. 11, pp. 1617–1623, Nov. 2022, number: 11 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41587-022-01432-w

[176] A. Madani, B. Krause, E. R. Greene, S. Subramanian, B. P. Mohr, J. M. Holton, J. L. Olmos, C. Xiong, Z. Z. Sun, R. Socher, J. S. Fraser, and

N. Naik, "Large language models generate functional protein sequences across diverse families," *Nature Biotechnology*, vol. 41, no. 8, pp. 1099–1106, Aug. 2023, number: 8 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41587-022-01618-2

[177] J. Horne and D. Shukla, "Recent Advances in Machine Learning Variant Effect Prediction Tools for Protein Engineering," *Industrial & Engineering Chemistry Research*, vol. 61, no. 19, pp. 6235–6245, May 2022, publisher: American Chemical Society. [Online]. Available: https://doi.org/10.1021/acs.iecr.1c04943

[178] H.-X. Yu, J. Wu, and L. Yi, "Rotationally Equivariant 3D Object Detection," Jul. 2022, arXiv:2204.13630 [cs]. [Online]. Available: http://arxiv.org/abs/2204.13630

[179] R. Roche, B. Moussad, M. H. Shuvo, S. Tarafder, and D. Bhattacharya, "EquiPNAS: improved protein-nucleic acid binding site prediction using protein-language-model-informed equivariant deep graph neural networks," Sep. 2023, pages: 2023.09.14.557719 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2023.09.14.557719v1

[180] M. Mirdita, K. SchÃ¼tze, Y. Moriwaki, L. Heo, S. Ovchinnikov, and M. Steinegger, "ColabFold: making protein folding accessible to all," *Nature Methods*, vol. 19, no. 6, pp. 679–682, Jun. 2022, number: 6 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41592-022-01488-1

[181] P. J. Ballester and W. G. Richards, "Ultrafast shape recognition to search compound databases for similar molecular shapes," *Journal of Computational Chemistry*, vol. 28, no. 10, pp. 1711–1723, Jul. 2007.

[182] R. Alapati, M. H. Shuvo, and D. Bhattacharya, "SPECS: Integration of side-chain orientation and global distance-based measures for improved evaluation of protein structural models," *PLOS ONE*, vol. 15, no. 2, p. e0228245,

Feb. 2020, publisher: Public Library of Science. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0228245

[183] R. Roche, B. Moussad, M. H. Shuvo, and D. Bhattacharya, "E(3) equivariant graph neural networks for robust and accurate protein-protein interaction site prediction," *PLOS Computational Biology*, vol. 19, no. 8, p. e1011435, Aug. 2023, publisher: Public Library of Science. [Online]. Available: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1011435

[184] R. Roche, S. Bhattacharya, M. H. Shuvo, and D. Bhattacharya, "rrQNet: Protein contact map quality estimation by deep evolutionary reconciliation," *Proteins*, Jun. 2022.

[185] M. Bertoni, F. Kiefer, M. Biasini, L. Bordoli, and T. Schwede, "Modeling protein quaternary structure of homo- and hetero-oligomers beyond binary interactions by homology," *Scientific Reports*, vol. 7, no. 1, p. 10480, Sep. 2017, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41598-017-09654-8