Accurate Conformation Sampling via Protein Structural Diffusion

Jiahao Fan[1,2†], Ziyao Li[2,3†], Eric Alcaide[2,4], Guolin Ke[2*], Huaqing Huang[1*], Weinan E[5*]

[1]School of physics, Peking University, Beijing, 100871, China.
[2]DP Technology, Beijing, 100080, China.
[3]Center for data science, Peking University, Beijing, 100871, China.
[4]University of Barcelona, Barcelona, Spain.
[5]School of Mathematical Sciences, Peking University, Beijing, 100871, China.

*Corresponding author(s). E-mail(s): kegl@dp.tech; huaqing.huang@pku.edu.cn; weinan@math.pku.edu.cn;
Contributing authors: jiahaofan@pku.edu.cn; leeeezy@pku.edu.cn;
†These authors contributed equally to this work.

Abstract

Accurately sampling of protein conformations is pivotal for advances in biology and medicine. Although there have been tremendous progress in protein structure prediction in recent years due to deep learning, models that can predict the different stable conformations of proteins with high accuracy and structural validity are still lacking. Here, we introduce Diffold, a cutting-edge approach designed for robust sampling of diverse protein conformations based solely on amino acid sequences. This method transforms AlphaFold2 into a diffusion model by implementing a conformation-based diffusion process and adapting the architecture to process diffused inputs effectively. To counteract the inherent conformational bias in the Protein Data Bank, we developed a novel hierarchical reweighting protocol based on structural clustering. Our evaluations demonstrate that Diffold outperforms existing methods in terms of successful sampling and structural validity. The comparisons with long time molecular dynamics show that Diffold can overcome the energy barrier existing in molecular dynamics simulations and perform more efficient sampling. Furthermore, We showcase Diffold's utility in drug discovery through its application in neural protein-ligand docking. In a blind test, it accurately predicted a novel protein-ligand complex, underscoring its potential to impact real-world biological research. Additionally, we present other modes of sampling using Diffold, including partial sampling with fixed motif, langevin dynamics and structural interpolation.

1 Introduction

Proteins rely on their three-dimensional structures to participate in biological processes, from catalyzing metabolic reactions to mediating cellular signaling. Recent deep learning advancements, epitomized by AlphaFold2[1], have revolutionized protein structure prediction. However, proteins are not static in vivo, they dynamically adopt multiple conformations, enabling interactions with diverse binding partners across distinct states. And *sampling diverse, low-energy conformations of proteins* remains an open challenge. Advanced computational approaches to capture this structural heterogeneity hold profound implications for both biological research and medical applications. As an example, through conformational sampling, researchers can identify transient structural pockets on the surfaces of proteins [2], revealing potential targets for therapeutic interventions that are not discernible in static structural representations. Traditional simulation methods mainly rely on molecular dynamics (MD) [3], yet they struggle with the protein's intrinsic complexity and high free energy barriers, making it difficult to capture essential dynamics on realistic timescales. While enhanced sampling methods have made strides [4–6], the timescale required to capture the essential dynamics and conformational changes is still hard to achieve in general.

Recent studies show that with smaller subsampled mutiple sequence alignments (MSAs) or selective MSA clusters as inputs, AlphaFold2 is capable of yielding alternate protein conformations [7, 8]. However, the potential bias introduced by the targeted MSA reduction is yet to be studied. For instance, we find in our experiments that such methods either cannot sample the known structures or do not have high sampling precision. Typical results from MSA clustering, or MSA subsampling with shallow MSA, are that a wide variety of conformations are generated, yet the majority are distant from any known structures. Although the pLDDT metric can be used as a confidence score to select the useful conformations,

thermodynamically disfavoured fold-switching (FS) state generated by MSA clustering was found to have a higher pLDDT than the ground state [8]. So we argue that a good sampler should mainly sample the low-energy conformations.

An alternative approach to conformation sampling is through generative modeling. Unlike MSA subsampling or MSA clustering methods which manipulate the input on trained AlphaFold2 model, generative modeling performs training on resolved protein structures and uses the trained model to sample structures in an unbiased way with only amino acid sequence and noise as input. The rationale behind generative models is that distinct resolved structures with the same sequence exist in the training set, so a structural probability distribution on amino acid sequences is expected to be learned by the model. Unlike molecular dynamics (MD) methods, which inherently perform correlated sampling, generative models facilitate direct sampling after training, thereby circumventing the free energy barriers characteristic of MD techniques. Recent investigations have explored the potential of diffusion- or flow-based models combined with AlphaFold to sample a diverse range of protein conformations [9, 10]. Despite the innovative nature of this approach, our findings indicate that current implementations fall short in terms of sampling accuracy or structural validity. Additionally, it is important to recognize alternative research directions that employ generative models for protein structure generation, primarily focusing on protein design rather than structural sampling [11, 12].

To evaluate the performance of different sampling methods, we hypothesized that chains with rather different structures but with the same sequence in the PDB are good target cases — 47 chains are selected in our evaluation. The rationale behind the testset is twofold: Firstly although protein chains often coexist with other chains as multimers or with other ligands as complex, the structures they adopt in the complex are generally in low-energy states stabilized by the neighboring chains or ligands. Secondly, in a practical viewpoint, when used as targets for rigid docking, the sampled structures of the chains are better in the conformation existing in the complex rather than in their alone to get better results. Thus rather than sampling different conformations of the chains where they adopt in their alone, we argue that a good sampling method could sample conformations of proteins in low-energy states either in their alone or in the complex they could form.

To fully tap the potential of deep generative models in conformation sampling, we propose Diffold, a structural diffusion model based on fine-tuning AlphaFold2. Specifically, we constructed an explicit diffusion process over each protein residue's position and rotation angle space as in Fig. 1a, and use a modified AlphaFold2 model to predict protein structures given diffused structures at different noise scale as input. As shown in Fig. 1b, we introduce several new modules and modifications to AlphaFold2, both to fully utilize the pre-trained parameters and to accommodate the diffused structural inputs. Briefly speaking, we remove the structure template channel in the AlphaFold2 model, and add a channel to receive the time embedding representing the noise scale. We reuse the channel receiving the initial structure in the structural module in AlphaFold2 model, to receive the diffused structures in Diffold, the details of the model architecture can be found in 4.3. Diffold is trained with continuous noise scheduling, which allows sampling with an arbitrary number of diffusion steps during inference, thus introducing a natural tradeoff between conformation quality and running time. During inference, a noisy structure is sampled randomly, and Diffold is run iteratively to refine the noisy structures to the predicted structures. Additionally, we employ a novel random span-masking strategy during training, which enables partial sampling of conformations with fixed motifs. Most critically, we develop a new hierarchical reweighting protocol of training samples based on structural clustering, which significantly mitigates the bias of conformation distributions in the Protein Data Bank (PDB) [13].

When evaluated on the curated benchmark, Diffold outperforms existing methods in terms of successful sampling and structural validity. Further comparisons with MD simulations show that Diffold can overcome the free energy barrier inherited in the MD methods and perform more efficient sampling. We also show through a case study that the conformations generated by Diffold could improve the performance of the docking method in a blind and cross docking setup, successfully predict the final docking pose ($\leq$ 2.0 Å). We also introduce three novel methods of manipulating Diffold beyond direct sampling, namely *Partial sampling with fixed motif*, *Langevin dynamics* and *structural interpolation*. The capability of Diffold could enable characterization of multiple conformations of unseen proteins, and aid in the discovery of targeted binding molecules.

Code and service of Diffold will be available upon publication.

2 Results

2.1 Training and Inference

Diffold was implemented based on the open-source training code of Uni-Fold [14], and the model parameters were initialized from the newest pre-trained weights of AlphaFold-Multimer (v3), released on 2022-12-06 (the training date cutoff is 2018-04-30). All protein chains in the Protein Data Bank released before 2022-04-30 were used in fine-tuning, while no AlphaFold-generated structures (i.e. self-distillation samples) were used. We resorted to Uni-Fold [14] for generated multiple sequence alignments (MSAs), which followed the MSA curation protocol of AlphaFold. We did not use structural templates. It took 16 NVIDIA A100 GPUs with 80GB memory for approximately 84 hours to fine-tune the model, with a batch size of 64 and a total of 20,000 steps. More details of model training are introduced in A.5. In the inference stage, we generated MSAs with the online server of MMSeqs2 [15], which adopted an MSA search protocol described in ColabFold [16]. Unless otherwise specified, the reverse processes were discretized uniformly to 30 steps.

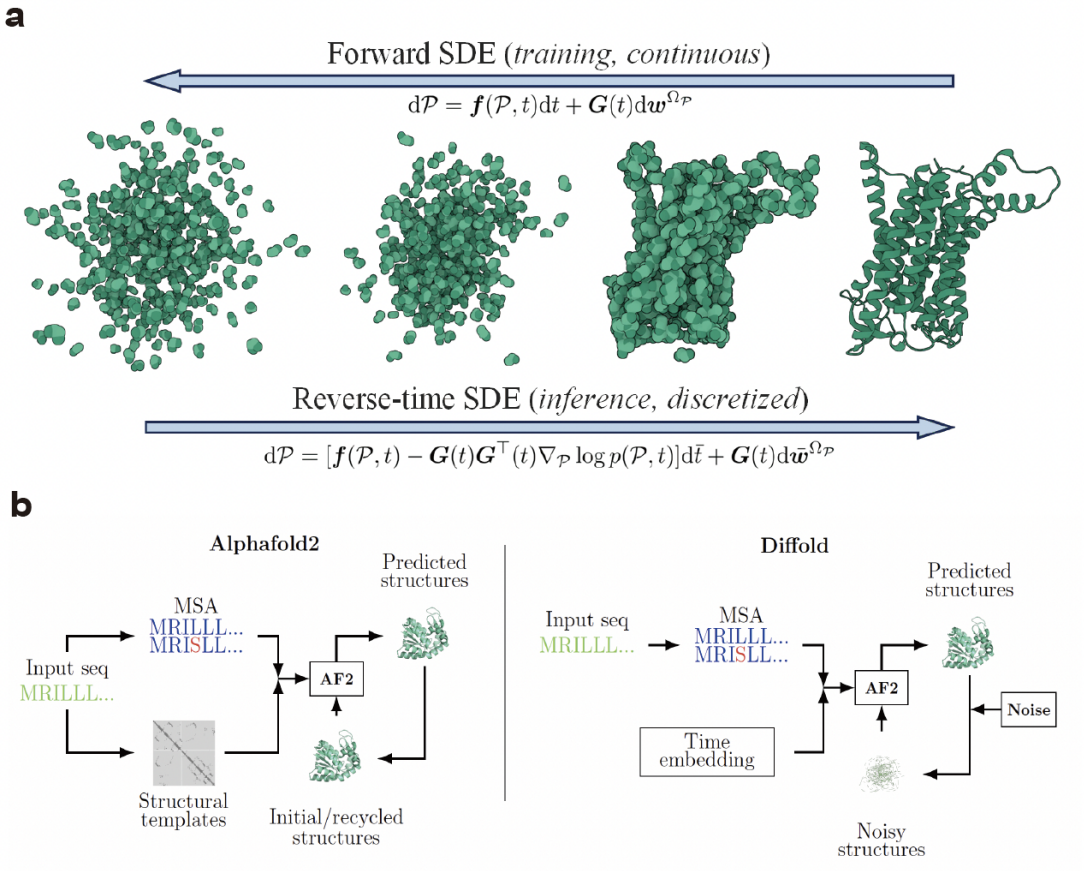


Fig. 1: Illustration of structural diffusion and overview of Diffold architecture.(a) Forward and backward structural diffusion process for proteins. During the training phase, Diffold is trained to recover the true structures with different scale of noise added on. During the inference phase, Diffold is run iteratively by reversing the diffusion process from random noise to produce structures for a given sequence. (b) Diffold architecture is modified from AlphaFold2, with structural templates channel removed, and time embedding channel added to represent the input noise scale. The initial/recyvled structures channel is reused in Diffold to receive the noisy structures. Diffold is trained by fine-tuning the trained AlphaFold2 model.

2.2 Sampling Alternative Conformations

We evaluated the ability of Diffold in sampling alternative protein conformations on a curated benchmarks, namely RAC-47, in comparison with AF-cluster [8] (an MSA clustering-based method to generate multiple conformations), MSA-subsampling [7] (different protein conformations are generated by MSA subsampling) and AlphaFlow [10] (a recent flow-matching model based on AlphaFold). We also compare the results with DiG [9] (a diffusion-based model using the Graphormer network [17]) through case study, since the authors only disclose the input feature data files for cases in their paper. Furthermore, Diffold is compared with MD simulation on selected cases, where different experimental structures are chosen as starting conformation for MD simulation.

Benchmark performance.

We evaluated the performance of different methods by curating a new benchmark, **RAC-47**, abbreviated from *47 recent proteins with alternative conformations*. Specifically, we collected all PDB chains released between 2022-04-30 and 2024-03-29 and selected sequence clusters containing these chains with 100% identity and have 2-10 chains in each cluster. We further filtered the clusters with sequence length between 128 and 768, and the maximum RMSD between chains in the clusters > 2 Å, resulting in 47 pairs of chains. The dataset generation process is unbiased and the details can be found in B.1. We sampled 100 conformations with Diffold, AlphaFlow [10], AF-cluster [8] and MSA-subsampling [7] methods for each of the pairs. Specifically, during inference by Diffold, we subsample the MSA with depth 1024 for each new sample, but not for each inference step, which is the same with the default setup for AlphaFlow. For AF-cluster, we select the first 100 MSA clusters (or all clusters if the cluster number is smaller than 100) generated by its open-source implementation, and ran AlphaFold on each. For MSA-subsampling, we run inference by subsampling MSA with depth 64, 256, and 1024. The result for subsampling MSA with depth 1024 is the same with the default setup for Alphafold. The same raw MSAs are used for all models. We reported the best TM-score of the sampled 100 conformations to the 2 real conformations. Moreover, to evaluate both the diversity and accuracy of the generated models, we used the *matchness score* [18] of precision and recall

-	Mean *Diff.*	Alphafold	Diffold	AlphaFlow
Mean-of-Best TM-score (↑)	0.753	0.811/0.783	0.839/0.824	**0.850**/**0.826**
Median-of-Best TM-score (↑)	0.757	0.866/0.808	0.915/**0.896**	**0.916**/0.876
Num. of successful sampling (↑)	-	9	**19**	18
mean/median MAT-R (↑)	-	0.797/0.817	0.832/**0.892**	**0.838**/0.884
mean/median MAT-P (↑)	-	0.800/0.829	0.760/0.790	0.797/0.834

-	MSA-subsampling (64)	MSA-subsampling (256)	AF-cluster
Mean-of-Best TM-score (↑)	0.818/0.793	0.822/0.794	0.696/0.688
Median-of-Best TM-score (↑)	0.863/0.809	0.856/0.807	0.722/0.722
Num. of successful sampling (↑)	11	12	2
mean/median MAT-R (↑)	0.805/0.839	0.808/0.841	0.692/0.734
mean/median MAT-P (↑)	0.750/0.771	0.795/0.806	0.492/0.504

Table 1: Evaluation results on RAC-47. Mean *Diff.* indicates the mean TM-score between two real conformations across 47 cases. The best TM-score of 100 generated conformations to both of the real ones in each case are reported and aggregated. The depth of MSA for MSA-subsampling model is indicated in the bracket.

(MAT-P / MAT-R) to assess the overall quality of all generated conformations, which was originally designed to measure qualities of molecular conformations. Specifically,

$$\text{MAT-P}(\mathcal{R}, \mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{\hat{\mathcal{P}} \in \mathcal{G}} \max_{\mathcal{P} \in \mathcal{R}} d(\hat{\mathcal{P}}, \mathcal{P}), \tag{1}$$

$$\text{MAT-R}(\mathcal{R}, \mathcal{G}) = \frac{1}{|\mathcal{R}|} \sum_{\mathcal{P} \in \mathcal{R}} \max_{\hat{\mathcal{P}} \in \mathcal{G}} d(\hat{\mathcal{P}}, \mathcal{P}), \tag{2}$$

where $\mathcal{R}$ and $\mathcal{G}$ are the sets of real and generated conformations, and we set the metric d as the TM-score between two conformations. Intuitively, MAT-P measures the average precision of the generated conformations concerning the closest real conformation, and MAT-R measures how well the real conformations are covered by the generated ones.

Table 1 reports the performance of all models on **RAC-47**. Specifically, we reported the mean and median of the best TM-score among 100 generated conformations corresponding to two real conformations across the 47 cases. Detailed results for individual cases can be seen in B.2. From Table 1 we can see that Diffold performs the best in terms of the median TM-score, while AlphaFlow achieves the best mean TM-score. To evaluate different models in terms of how successful the sampling is, we defined a "successful sampling" to be that each of the best TM-score for a given case is larger than 0.90. This is a rational metric considering that the relative TM-score between two experimental conformations of all cases in **RAC-47** ranges from 0.461 to 0.900. From Table 1 we can see that Diffold achieves the most successful sampling with 19 among 47 cases. Additionally, we reported the average MAT-P and MAT-R across the 47 cases. Regarding recovery, Diffold and AlphaFlow performs the best in terms of median and mean MAT-R respectively. In terms of MAT-P, Diffold outperform the MSA-subsampling with MSA depth 64 and AF-cluster, while behind Alphafold, AlphaFlow and MSA-subsampling with MSA depth 256. The superiority of Diffold can also be seen in Fig. 2a-h, where two typical sampling cases are shown. We could see that compared with AlphaFlow, the sampling of Diffold is more diverse, thus explaining in part why AlphaFlow outperform Diffold in MAT-P. But the diversity of the sampling in Diffold is controlled so that the generated conformations do not deviate from the experimental structures too much, so the diverse sampling exhibited by Diffold might be advantageous in situations where rather different conformations are needed. On the other hand, the diversity of the sampling in MSA-subsampling and AF-cluster are generally uncontrolled, thus leading to mostly unwanted conformations.

From the above analysis we can see that Diffold outperform MSA-subsampling and AF-cluster methods in terms of sampling recall and displaying reasonable sampling precision, while having comparable performance with AlphaFlow. To further evaluate these two models, it should be noted that structural validity must be taken account. To evaluate whether the generated configurations satisfy the physical constraints induced by the physical law, we calculated the violation loss and different contributions to it according to the established work[1, 19] for the generated conformations across 47 cases in RAC-47 for the best two models so far: Diffold and AlphaFlow. The results from Alphafold was also calculated as comparisons.

As shown in Fig. 2i-l, Diffold exhibited an equal or superior performance with Alphafold on both the total violation loss metric and also different contributions to it, including the C-N bond loss, clash counts between residues, and the clash counts within residues. The clash counts is defined to be the number of atom pairs whose distance is smaller than the atomic van der waals radius with a tolerance. On the other hand, the violation loss and each contributed loss in AlphaFlow are significantly higher than those of Diffold and Alphafold, which indicated that the conformations generated by AlphaFlow have physical invalidity.

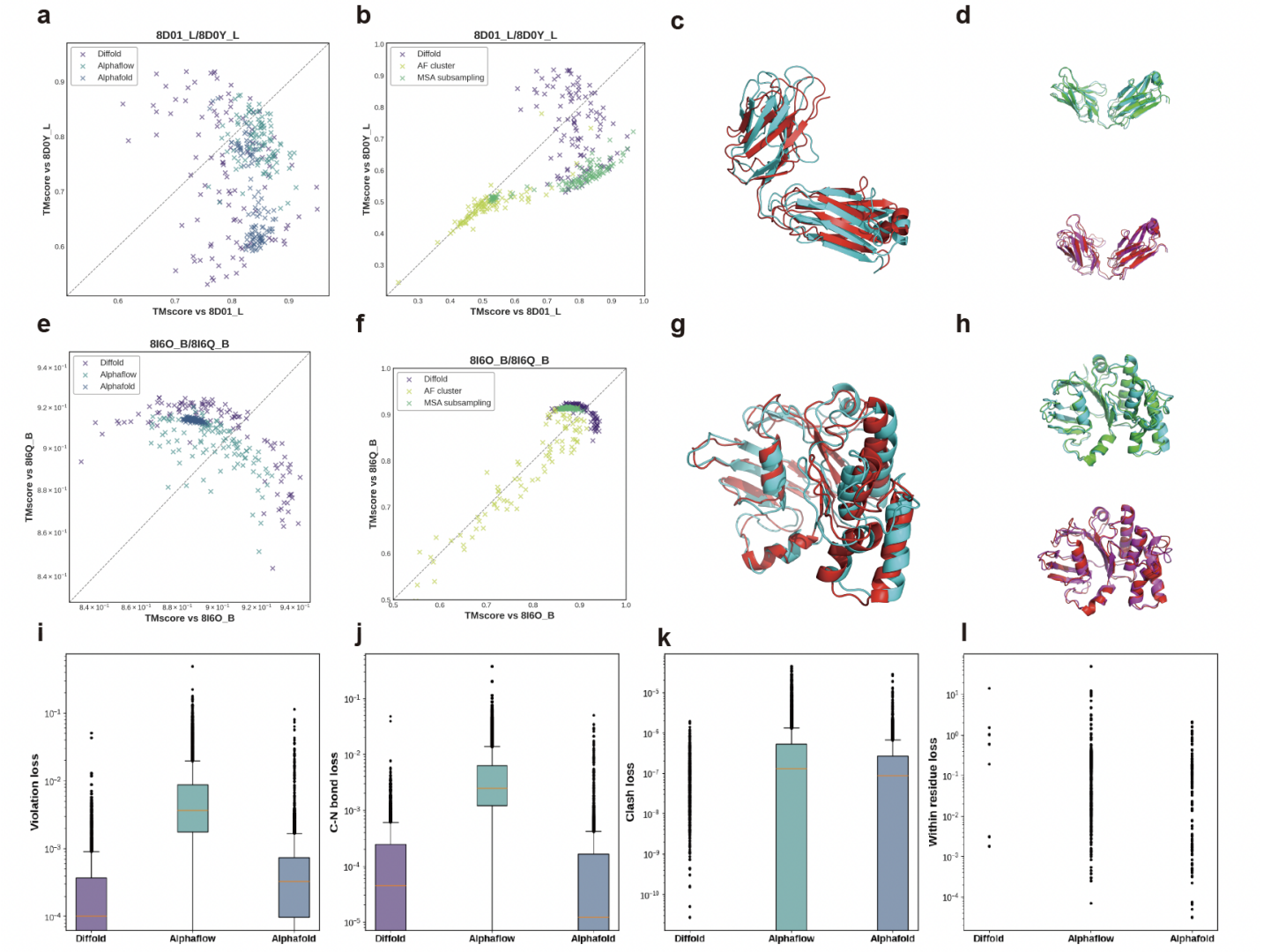


Fig. 2: Performance comparison of different models on selected cases and comparison of structural violations between different models. (a)(b) Scatter plot of TM-score between 8D01_L/8DOY_L and 100 conformations generated by five different models. (c)(d) Overlay of sampled structures from Diffold with 8D01_L/8DOY_L. cyan: 8D01_L experimental structure; red: 8DOY_L experimental structure; green: sampled structure closest to 8D01_L; magenta: sampled structure closest to 8DOY_L. (e)(f) Scatter plot of TM-score between 8I6O_B/8I6Q_B and 100 conformations generated by five different models. (g)(h) Overlay of sampled structures from Diffold with 8I6O_B/8I6Q_B. cyan: 8I6O_B experimental structure; red: 8I6Q_B experimental structure; green: sampled structure closest to 8I6O_B; magenta: sampled structure closest to 8I6Q_B.(i) Total violation loss as defined in [19] for all generated conformations; (j) Carbon-Nitrogen(C-N) bond loss (indicating the violation from the C-N bond length) for all generated conformations; (k) Clash counts between residues (indicating the violations from the atomic radius restriction between residues) for all generated conformations; (l) Clash counts within residues (indicating the violations from the atomic radius restriction within residues) for all generated conformations;

Comparison with DiG and MD.

To compare Diffold with DiG[9], we selected cases described in their papers since they only disclose the input files for those cases in their open-sourced code. As shown in Fig. 3a-d, we can see that the sampling of DiG is concentrated on a small region for both cases, while Diffold have a balanced sampling between 1AKE_A and 4AKE_A conformations, and have a more diverse sampling on 2DRI_A/1URP_A case than DiG. Furthermore, it should be noted that conformations generated by DiG only contain the backbone of the proteins without side chains, while all other prediction methods produce the all-atom protein structures.

Besides comparing Diffold with different machine learning based methods, we present a comparative analysis of our model Diffold against results from molecular dynamics (MD) simulations. The MD simulations were conducted over a

duration of 100 ns. Due to the substantial computational demands associated with prolonged MD simulations, we limit our comparison to selected cases. As illustrated in Fig. 3e-j, the MD simulations fail to explore alternative conformations from a starting conformation in the 8DKE_P/8DKI_P pairs, whereas Diffold successfully samples both conformations within 100 samples. Notably, while MD simulations can transition from the 7SJO_F to the 7SJP_L conformation, they do not reciprocate this transition, suggesting a potential stability preference for the 7SJP_L conformation. In contrast, Diffold demonstrates the capability to sample both conformations effectively within 100 iterations.

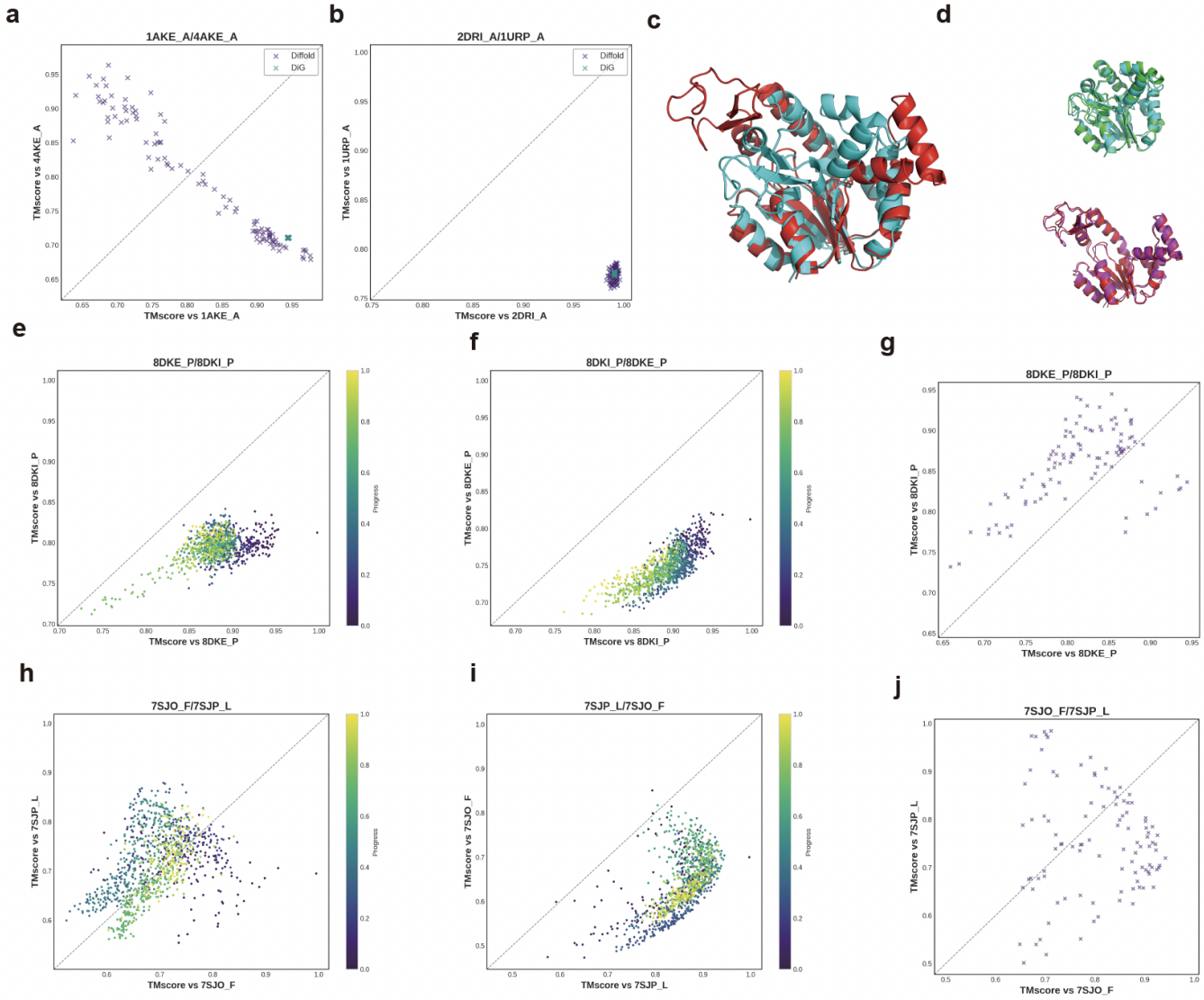


Fig. 3: Performance comparisons of Diffold with DiG and with MD simulations. (a) Scatter plot of TM-score between 1AKE_A/4AKE_A and 100 conformations generated by Diffold and DiG. (b) Scatter plot of TM-score between 2DRI_A/1URP_A and 100 conformations generated by Diffold and DiG. (c)(d) Overlay of sampled structures from Diffold with 1AKE_A/4AKE_A. cyan: 1AKE_A experimental structure; red: 4AKE_A experimental structure; green: sampled structure closest to 1AKE_A; magenta: sampled structure closest to 4AKE_A. (e-f) Scatter plot of TM-score between 8DKE_P/8DKI_P and 100 conformations generated by MD simulations, where the MD simulations are start from 8DKE_P in (e), and start from 8DKI_P in (f), the colorbar indicates the simulation progress. (g) Scatter plot of TM-score between 8DKE_P/8DKI_P and 100 conformations generated by Diffold.(h-i) Scatter plot of TM-score between 7SJO_F/7SJP_L and 100 conformations generated by MD simulations, where the MD simulations are start from 7SJO_F in (h), and start from 7SJP_L in (i). (j) Scatter plot of TM-score between 7SJO_F/7SJP_L and 100 conformations generated by Diffold.

Ablation study

To assess the impact of our proposed hierarchical reweighting protocol based on structural clustering in the training dataset, and also the discretization steps during inference, we trained an additional version of Diffold using AlphaFold's

sequence clustering-based sampling schema, where sample weights were determined by the inverse sequence cluster size and conformations with identical sequences were uniformly sampled. All experiments were conducted on RAC-47, with the mean MAT-P and MAT-R results summarized in Table 2. We find that the model can produce fairly good predictions with even 1 step from the prior. The precision of the model was impacted with 10 steps, as errors were not negligible under such coarse discretizations. Adding inference steps can improve recall performance, though gains marginalize after 30 steps (whose results are reported as default). Removing the hierarchical reweighting protocol leads to degraded precision and recall. The results corroborate the importance of the reweighting protocol.

#samples	Reweighting	#steps	mean MAT-P (↑)	mean MAT-R (↑)
100	Yes	1	0.767(+0.07)	0.815(-0.17)
	Yes	10	0.750(-0.10)	0.820(-0.12)
	Yes	30	0.760(+0.00)	0.832(+0.00)
	Yes	50	0.760(+0.00)	0.831(-0.01)
	No	30	0.722(-0.38)	0.801(-0.31)
5	Yes	30	0.755(+0.00)	0.772(+0.00)
	No	30	0.727(-0.28)	0.733(-0.39)

Table 2: Impacts of the number of inference steps and the hierarchical reweighting protocol on RAC-47.

2.3 Case Study

Recent research [20] has evidenced the dynamic nature of SLC15A4 and the crucial role its multiple conformations play in the TLR-IRF5 immune signaling, whose misregulation thought to be crucial in *systemic lupus erythematosus* (SLE) and related diseases. Specifically, the TASL binds to SLC15A4 in its cytoplasmic inward-open confromation (PDB ID: 8JZU), participating in the nucleic acid-sensing TLR7/8 pathway activating IRF5. While the newly characterized inhibitory molecule *feeblin* acting as an allosteric inducer, stablizes SLC15A4 in a TASL binding-incompetent lysosomal outward-open conformation (PDB ID: 8JZX), leading to degradation of TASL. To showcase the usage of conformation sampling for accurate modeling of multi-state proteins, we apply Diffold in a blind setup to SLC15A4 and show its importance for accurately characterizing the newly discovered inhibitor-binding mode. It is important to note that the two experimental conformations are out of the training set of Diffold.

Before demonstrating our method, it should be noted that in a easier redocking setup where the experimental structure (8JZX) is used as the target for docking, existing well-known docking methods like quickvina-W, smina and diffdock [21–23] can not find the correct (ligand RMSD $\leq$ 2.0 Å) docking pose, the details are given in B.3. We now turn to illustrate our method in a more challenging setup where the protein structure information is completely unknown.

The protocol consisted of sampling 100 conformations with Diffold. From Fig. 4e we can see that both cytosol-open (8JZU) and lysosome-open conformations (8JZX) could be sampled (TMscore > 0.9). Pockets for all conformations were identified with FPocket 4.0 [24] and default parameters and correlations to the TASL binding pocket were investigated. Interestingly, the volume of a central pocket accessible from the lysosome side was found to correlate with protein conformation and inversely with TASL in volume, so it was selected for docking of *feeblin*. A ligand conformation and predicted confidence (pRMSD) were produced for each conformation using the UniMol Docking and UniDock tools [25, 26]. Prediction confidence was smoothed by KNN (K=3) over predicted ligand conformers. From Fig. 4f we can see that the predicted RMSD is correlated with the true RMSD of the ligand (*Spearman* ρ(pRMSD, RMSD) $= 0.74$, $p < 10^{-17}$) and with the protein conformations (*Spearman* ρ (pRMSD,TM-8JZU - TM-8JZX) $= 0.50$, $p < 10^{-6}$). The prediction confidence and the actual performance is generally better when the predicted conformations are closer to lysosome-open conformation (8JZX). The best prediction by smoothed confidence was taken, which obtains $\leq$ 2.0 Å ligand RMSD to the experimental structure (PDB ID: 8JZX).

Notably, as shown in Fig. 4g, MD simulation from 8JZU can not sample 8JZX conformation during 100ns run, indicating an energy barrier between the two conformations. This example demonstrates the utility of sampling multiple protein conformations using Diffold to improve the docking performance, thus illuminating the usage of the model in real drug discovery scenarios.

2.4 Partial Sampling, Langevin Dynamics and Structural Interpolation

Diffold can be used to perform partial sampling of proteins with fixed motif, which is illustrated in one example in RAC-47 benchmark. We selected 7UTI_Z/7UTL_d case since the sampling performance from Diffold is bad, as shown in Fig. 5b. Further investigation indicated that it is mainly due to the flexible loop between the two domains in the protein which leads to bad performance. As shown in Fig. 5a, the middle loop has large diversity, the loop direction of a typical sampled structure (green) from Diffold deviates a lot from the two experimental structure (cray and red), thus leading to large structural disimilarity. We hypothesized that fixing the loop direction aligned with the experimental structures will lead to

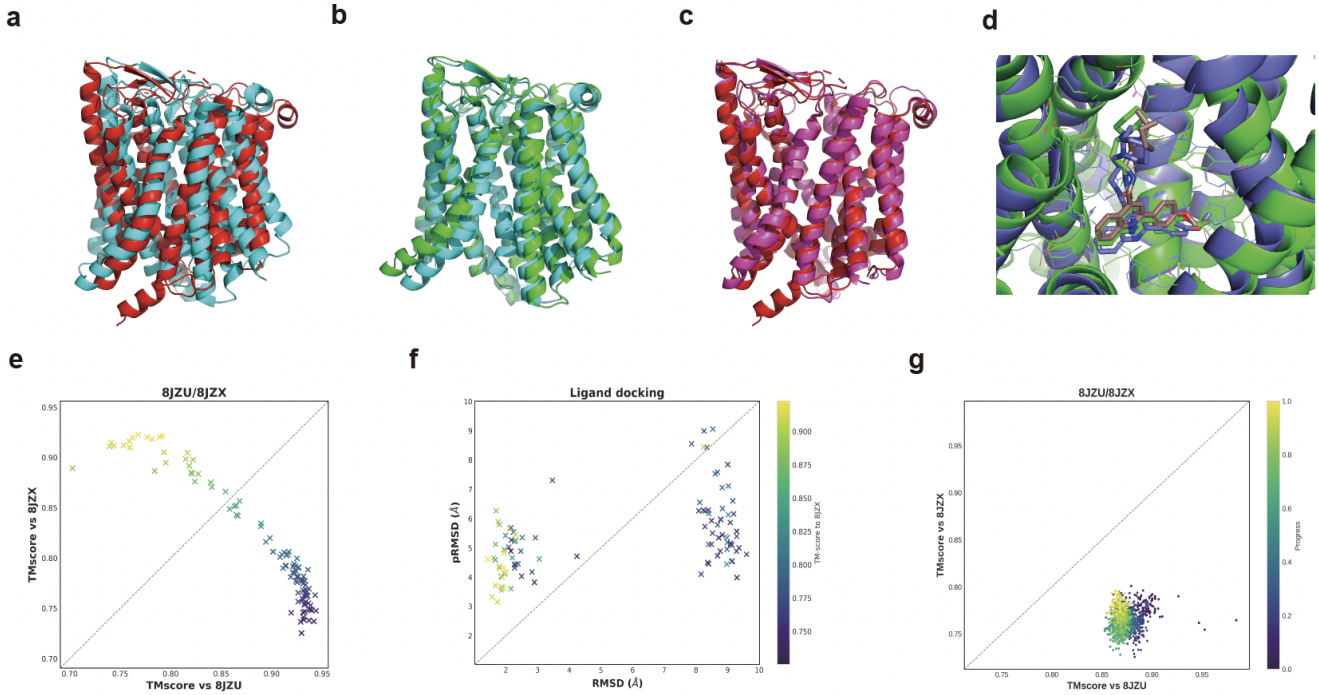


Fig. 4: Visualization and scatter plots of SLC15A4 conformations and binding poses. (a-c) Overlay of sampled structures from Diffold with 8JZU/8JZX. cyan: 8JZU experimental structure; red: 8JZX experimental structure; green: sampled structure closest to 8JZU; magenta: sampled structure closest to 8JZX. (d) Visualization of binding site. green: 8jzx experimental structure; brown: ligand bound to experimental structure; blue: predicted ligand and protein structure. (e) Scatter plot of TM-score between the 100 generated conformations and 8JZX/8JZU.(f) Scatter plot of ligand predicted RMSD v.s. RMSD using each sampled conformation, the colorbar indicates the TM-score to 8JZX where the same color corresponds to the same conformation to (e). (g) Scatter plot of TM-score between 8JZU/8JZX and 100 conformations generated by 100ns MD simulations, where the MD simulations are start from 8JZU. The colorbar indicates the simulation progress.

increase in the sampling accuracy. The results showed that, when using the left domain and the loop region from the 7UTI_Z (cray) experimental structure as fixed motif, the sampled structures will be close to 7UTI_Z with reasonable structural diversity as shown in Fig. 5c. On the other hand, when using the left domain and the loop region from the 7UTL_d (red) experimental structure as fixed motif, the sampled structures will be close to 7UTL_d with reasonable structural diversity as shown in Fig. 5d.

Sometimes we not only want to sample low-energy structures, but also want to see the continuous transition between two stable structures. This could be done using Diffold in the framework of Langevin dynamics and structural interpolation. We illustrate the effects of Langevin dynamics and structural interpolation on adenylate kinase, a classic case of alternative conformations in apo and holo states. This nucleoside monophosphate kinase exhibits an open apo conformation (PDB ID: 4AKE) while adopting a closed conformation (PDB ID: 1AKE) when bound to an inhibitor [27]. We conduct 5000 steps of Langevin dynamics on the diffused conformations of $t = 0.6$, by iteratively integrating the forward and reverse processes with time step $\tau = 0.01$. The starting point was generated with the forward SDE on a predicted structure with $t = 0.6$. For interpolation, we diffused both 1AKE and 4AKE to $t = 0.3$, interpolated the structures with 100 intermediate steps, and predicted results with the reverse-time SDE from t, discretized with step size 1 or 10. Fig. 5 visualizes the generated trajectories. From Fig. 5e and Fig. 5f, we observe switches of conformations between 1AKE and 4AKE and several stationary states. More details and discussions are in B.5. Fig. 5g and Fig. 5h show that the interpolation results in a putative, continuous transition pathway between the two conformations. Using a discretized inference process of 10 steps (h) leads to more vibration of the structure compared with the direct prediction (g), while the overall trend remains the same.

3 Discussions

After the revolutionary breakthrough in protein structure prediction by AlphaFold2, determining different conformations of the proteins is another harder problem with great significance. Success at the problem could deepen our understanding of the dynamic nature of proteins and provide great help in structure based drug discovery. Our results demonstrate that by utilizing the capability of AlphaFold2 and finetuning it to a diffusion model, and through a dataset reweighting

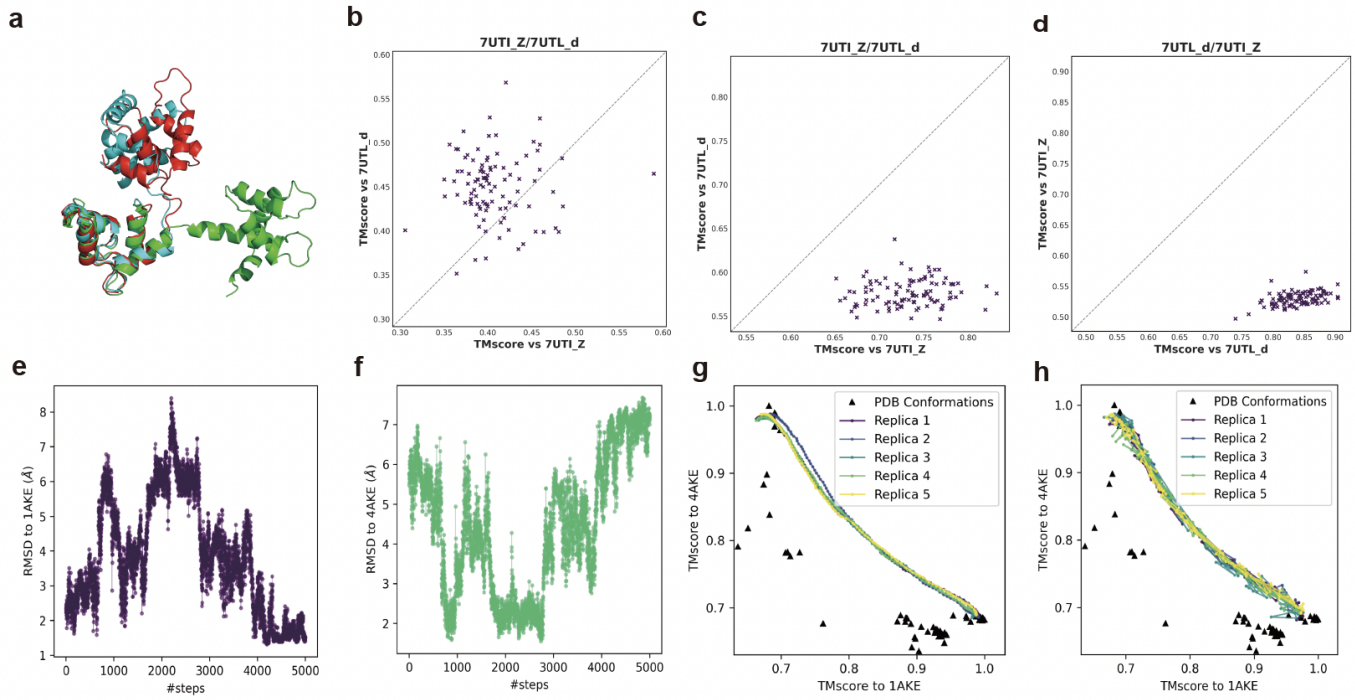


Fig. 5: Illustration of partial sampling, langevin dynamics and structural interpolation through case study. (a) Overlay of sampled structures from Diffold with 7UTI_Z/7UTL_d. cyan: 7UTI_Z experimental structure; red: 7UTL_d experimental structure; green: sampled structure from Diffold. (b) Scatter plot of TM-score between 7UTI_Z/7UTL_d and 100 conformations generated by Diffold. (c) Scatter plot of TM-score between 7UTI_Z/7UTL_d and 100 conformations generated by Diffold with left domain and loop region of 7UTI_Z fixed. (d) Scatter plot of TM-score between 7UTI_Z/7UTL_d and 100 conformations generated by Diffold with left domain and loop region of 7UTL_d fixed. (e-f) RMSDs of conformations in Langevin dynamics ($\hat{\mathcal{P}}^{(t)}$) to the two real conformations. (g-h) TMscores of interpolated conformations to the two real conformations, with 1 and 10 inference steps respectively. Other PDB structures in the kinase family are also shown.

protocal based on structural clustering, Diffold could achieve the best performance on curated benchmark RAC-47 among all compared models, successfully sample the most known protein chains with high structural validity.

We also compare Diffold with MD simulation results on selected cases, the results show that Diffold can overcome the energy barrier existing in MD simulations, thus sampling the stable conformations in a more efficient way. There are two possible reasons for the superior sampling efficiency of Diffold over MD simulations: Firstly it might be that it is mainly due to the direct sampling capability of the generative models which overcome the energy barrier existing between different low-energy states. Secondly it might be that it is due to the learned probability distribution itself by Diffold such that there is no free energy barrier or only low barrier existing between different states. Exploration of the possible reason is interesting in its own right and leaves as future work.

We further validated the utility of Diffold through a case study in blind and cross docking setups, demonstrating its ability to sample both cytosol-open and lysosome-open conformations from sequence input alone, thereby enhancing docking outcomes. We could see the whole pipeline of first using Diffold to sample conformations of a protein and then using docking methods to perform rigid docking as a way of doing ensemble docking. Since the prediction confidence of the docking method used could be an indicator to select the best binding pose, the whole pipeline can also be viewed as one way of flexible docking where the targeted protein structure could be changed.

It should be noted that we adopt a different viewpoint for what a good sampling model should be than previous works, where they claim that the sampled structures should be drawn from the boltzmann distribution or assigned a boltzmann weight [8, 9], characteristic of the canonical probability distribution. Firstly both AlphaFold2 and different generative models (Diffold, DiG, AlphaFlow) are mainly trained on resolved experimental structures, and these structures exist alone or with their binding partner. So the expectation of the trained models is to sample conformations of proteins in their alone or in the complex, unless the training dataset is changed from resolved experimental structures to the conformations from MD simulations. It is reasonable for an isolated protein to conform the canonical distribution, but in combination with the binding partners it is difficult to tell the distribution change of the proteins. So we hypothesize that a good sampling model should sample the conformations of proteins in low-energy states either in their alone or in the complex they could form, without the need to be drawn from a boltzmann distribution or assigned with the boltzmann weight unless there is specific needs to do so. While on the other hands, an indicator should indeed be assigned to the sampled structures as the stability or validity of the conformation, this might be done through calculation of the free energy of different conformations and serves as the future direction.

Another research direction is to extend the framework to protein multimers, whereas the challenge is as well to provide a balanced sampling protocol. Furthermore, we find in our experiment that by performing MSA clustering and using clustered MSA as Diffold input, the model could predict even more diverse structures, but we did not explore more on the combination of generative models and MSA manipulation methods which could also be interesting. We hope that beyond AlphaFold2, methods which could sample different conformations of proteins in low-energy states could provide us more understanding of the protein dynamics and functions.

4 Method

4.1 Preliminaries

AlphaFold2.

AlphaFold2 [1, 28] is a deep learning system that accurately predicts protein structures from amino acid sequences, multiple sequence alignments and structural templates. In summary, it introduces *Evoformer*, an attention-based network to encode input sequences and templates, and a *structure module*, an SE(3)-equivariant network to decode the predicted structure step-wise from the origin. A recycling mechanism is introduced to iteratively refine the predicted structure with the same network parameters. Subsequent works [14, 19] implemented open-source training code of AlphaFold2 to enable further development of the model. We implemented Diffold based on [14].

Notations and structure parameterization.

We build structural diffusion processes based on the parameterization of protein backbones introduced in AlphaFold2. Formally, a residue $\mathcal{A}_i$ in a protein structure $\mathcal{P} = (\mathcal{A}_1, \cdots, \mathcal{A}_n)$ is represented by $\mathcal{A}_i = (T_i, \mathcal{X}_i)$, where $T_i = (\boldsymbol{R}_i, \boldsymbol{c}_i) \in \mathrm{SE}(3)$ is the local frame of the residue, defined by the coordinates of the backbone atoms (N,C_α,C), and $\mathcal{X}_i = (\chi_1^i, \chi_2^i, \chi_3^i, \chi_4^i) \in \mathrm{SO}(2)^4$ are (up to) four torsion angles which fully determine the flexibility of the sidechain. We use n for the length of the protein, i, j for residue indices, and $\mathcal{S} = (s_1, \cdots, s_n)$ for the sequence of the protein.

Diffusion models.

Diffusion models [29, 30] have now become the mainstream method in computer vision. As probabilistic generative models, they characterize data generation as reverse-time diffusion processes. Further research [31] illuminated deep connections between diffusion models and score-based generative models. In a diffusion model, a stochastic diffusion process is created for $\boldsymbol{x} \in \Omega_{\boldsymbol{x}}$ as

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}^{\Omega_{\boldsymbol{x}}}, \tag{3}$$

where $\boldsymbol{f}(\cdot), g(\cdot)$ are pre-defined drift and diffusion factors, and $\mathrm{d}\boldsymbol{w}$ is the differential of a standard Wiener process. The reverse-time diffusion process takes the form of

$$\mathrm{d}\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g^2(t)\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}, t)\right] \mathrm{d}\bar{t} + g(t)\mathrm{d}\bar{\boldsymbol{w}}^{\Omega_{\boldsymbol{x}}}. \tag{4}$$

The training of the network aims to learn a parameterized score function $\boldsymbol{s}_\theta(\boldsymbol{x}, t)$ approximating $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}, t)$, which is only feasible (conditioned on data samples $\boldsymbol{x}_0 \in \mathcal{D} \subset \Omega_{\boldsymbol{x}}$) during model training. Recent work has extended diffusion models to protein domain for applications in protein design [11, 32, 33] and conformation sampling [9, 34]. When modeling the rotations and torsion angles of proteins, diffusion processes on Riemann manifolds [35] should be introduced.

Diffusion models on Riemann manifolds.

Riemannian diffusion models are extensions of Euclidean diffusion models which models the diffusion processes on Riemannian manifolds. In a Riemannian diffusion model, Wiener processes in Eq. (3) and (4) are replaced with Brownian motions on the Riemann manifold $\mathcal{M}$, and score functions $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ should be calculated as Riemannian gradients which lies in the tangent space $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$. We refer our readers to [35] and [12] for theories and more discussions. Our work adopts a similar definition of protein diffusion processes to [11], where the main differences are that torsion angles are incorporated and the forward process is not discretized during training.

4.2 Structural Diffusion

Fig. 1a illustrate the concept of structural diffusion. Overall, we create a diffusion process over the protein backbone space $\Omega_{\mathcal{P}} = \{\mathrm{SE}(3)\}^n$, and the components on position and rotation angles of each protein residue are defined independently. For simplicity, we hereby describe the diffusion process for a specific residue $\mathcal{A} \in \mathcal{P}$. For each backbone pose $T = (\boldsymbol{R}, \boldsymbol{c})$, we formulate an Euclidean diffusion process for the position and an SO(3) diffusion process for the rotation:

$$\mathrm{d}\boldsymbol{R} = \sqrt{\beta_{\mathrm{rot}}(t)}\mathrm{d}\boldsymbol{w}^{\mathrm{SO}(3)} \tag{5}$$

$$\mathrm{d}\boldsymbol{c} = -\frac{1}{2}\beta_{\mathrm{pos}}(t)\boldsymbol{c}\mathrm{d}t + \sqrt{\beta_{\mathrm{pos}}(t)}\mathrm{d}\boldsymbol{w}^{\mathbb{R}^3} \tag{6}$$

where $\mathrm{d}\boldsymbol{w}^*$s are differentials of Brownian motions in the corresponding spaces, and β_*s are pre-defined functions over t. Correspondingly, the marginal density of $(\boldsymbol{R}, \boldsymbol{c})$ at t, conditioned on $(\boldsymbol{R}^{(0)}, \boldsymbol{c}^{(0)})$, can be derived following [31] as

$$p(\boldsymbol{R}, t|\boldsymbol{R}^{(0)}) \propto \tilde{\psi}\left(\omega\left(\boldsymbol{R}\boldsymbol{R}^{(0)\top}\right); \sigma^2_{\mathrm{rot}}(t)\right), \tag{7}$$

$$p(\boldsymbol{c}, t|\boldsymbol{c}^{(0)}) = \mathcal{N}(\boldsymbol{c}; \sqrt{\gamma_{\mathrm{pos}}(t)}\boldsymbol{c}^{(0)}, (1-\gamma_{\mathrm{pos}}(t))\boldsymbol{I}), \tag{8}$$

where $\sigma^2_{\mathrm{rot}}(t) = \int_0^t \beta_{\mathrm{rot}}(s)\mathrm{d}s$ and $\gamma_{\mathrm{pos}}(t) = e^{-\int_0^t \beta_{\mathrm{pos}}(s)\mathrm{d}s}$ are integrated from $\beta(t)$. Here $\omega : \mathrm{SO}(3) \to [0, \pi)$ maps an SO(3) element to a rotation angle, and $\psi(\cdot)$ is the density function of rotation angles of isotropic Gaussian distribution on the SO(3) manifold (IGSO(3), Nikolayev et al. [36]), defined as

$$\tilde{\psi}(\omega; \sigma^2) = \frac{1-\cos\omega}{\pi}\psi(\omega; \sigma^2), \tag{9}$$

$$\psi(\omega; \sigma^2) = \sum_{l=0}^{\infty} e^{\left(-\frac{l^2+l}{2}\sigma^2\right)} \cdot \frac{\sin((l+\frac{1}{2})\omega)}{\sin(\omega/2)}. \tag{10}$$

The difference between $\tilde{\psi}$ and ψ is up to the volume density on the SO(3) manifold with regard to its Haar measure. The forms of the score functions can then be derived as

$$\nabla_{\boldsymbol{R}} \log p(\boldsymbol{R}, t|\boldsymbol{R}_0) = \frac{\mathrm{d}\psi}{\omega \mathrm{d}\omega}\mathrm{Log}(\boldsymbol{R}\boldsymbol{R}_0^\top),\ \omega = \omega(\boldsymbol{R}\boldsymbol{R}_0^\top), \tag{11}$$

$$\nabla_{\boldsymbol{c}} \log p(\boldsymbol{c}, t|\boldsymbol{c}_0) = \frac{\sqrt{\gamma_{\mathrm{pos}}(t)}\boldsymbol{c}_0 - \boldsymbol{c}}{1-\gamma_{\mathrm{pos}}(t)}, \tag{12}$$

where ψ and $\mathrm{d}\psi/\mathrm{d}\omega$ can be approximated numerically by truncating the summation to $l \leq L$.

The torsion angles are trained and generated in the same way with AlphaFold2 [1] though a shallow ResNet, thus do not undergo diffusion process. We tried to implement diffusion process on torsion angles, but the results were worse than the original model, this might be a research direction for future study.

In choosing the parameters of the diffusion process, we scaled the process in a continuous unit time interval $t \in [0, 1]$. We set a distorted *cosine schedule* [37] for the position diffusion and polynomial schedules for rotation and torsion diffusions. Specifically,

$$\gamma_{\mathrm{pos}}(t) = \frac{f(t)}{f(0)},\ f(t) = \cos^2\left(\frac{\pi}{2} \cdot \frac{t^a + s}{1+s}\right), \tag{13}$$

$$\sigma_{\mathrm{rot}}(t) = \sigma^{\max}_{\mathrm{rot}} \cdot t^b, \tag{14}$$

where a, b are chosen to empirically smooth the variance along the diffusion process. In positional diffusion, the center-of-mass of the protein is removed and the coordinates are normalized to 1/20.

Notably, instead of directly producing the estimated score functions, our model reparameterizes the score function by predicting $\hat{\mathcal{P}}_0^{(t)} \approx \mathcal{P}_0$. The estimated score function is then calculated by replacing elements in $\mathcal{P}_0$ in the above equations with those in $\hat{\mathcal{P}}_0^{(t)}$. This approach allows the utilization of intermediate model outputs as sampled protein conformations, as well as direct supervision of model outputs to enforce correct geometry. Detailed algorithms for model training and inference are provided in A.4.

To further allow partial sampling of protein conformations where a portion of the structure (motif) is fixed, we sample continuous motifs and fix them during training. Specifically, we propose a Markovian random span-masking strategy that generates motif masks with expected continuous segment size K and mask probability ρ. To generate a sequence of binary masks $(M_1, M_2, \cdots)$, we create

$$\begin{aligned} p_1 &= \rho, \qquad M_i \sim \mathrm{Bernoulli}(p_i),\ i \geq 1, \\ p_i &= \frac{K-1}{K}M_{i-1} + \frac{\rho}{(1-\rho)K}(1-M_{i-1}),\ i > 1. \end{aligned} \tag{15}$$

Properties of this span-masking are proved in A.1.

4.3 Architecture Modification

The motivation of all following modifications to the architecture of AlphaFold2 is to appropriately transform the protein folding model, $m_\theta(\mathcal{S})$, to a diffusion model, $m_\theta(t, \mathcal{P}^{(t)}, \mathcal{S})$. To achieve this, we introduced several new modules to Evoformer and altered the initial structure inputs to the *structure module* of AlphaFold2. We shut the template channel of AlphaFold2, as we wanted the input diffused structures to be the only source of structure information, thus dominating the outcomes of the model. A total of 1.6M trainable parameters were newly introduced, 1.7% to the original (92.9M, without template channels). Notably, to ensure that the model correctly utilizes the pre-trained parameters at the early stage of fine-tuning, we zero-initialized all outputs from the newly introduced modules.

Time embedding.

Diffold takes a continuous variable $t \in (0, 1]$ as input. This variable is first encoded with a radial basis function (RBF) and then transformed by a gated linear layer. Specifically, the gated linear layer was defined as $\boldsymbol{h} = \text{sigmoid}(\boldsymbol{W}_g \boldsymbol{x}) \odot (\boldsymbol{W}_l \boldsymbol{x})$, where $\boldsymbol{W}$s are network parameters and the $\odot$ represents element-wise multiplication. These time embedding features are properly reshaped, tiled by numbers of residues and MSAs, and added to the initial MSA and pair representations before inputting them to Evoformer. Notably, the time embedding of residues also indicates missing residues and fixed motifs by setting corresponding representations to $t = 0$ or 1 (see A.1).

Relative coordinates recycling.

In the recycling module, we removed the recycling of MSA and pair representations of AlphaFold2 but retained the recycling of predicted conformations. Specifically, we do not recycle predicted structures $\hat{\mathcal{P}}_0^{(t)}$, but instead use the module to encode pair-wise distances of the diffused structure $\mathcal{P}^{(t)}$. We additionally encode relative coordinates between residues, $\boldsymbol{r}_{ij} = \mathbf{T}_j^{-1} \circ \mathbf{c}_i$, as pair features. These features are processed with RBFs dimension-wise and 2 residual blocks [38], and then added to the pair representations as Evoformer inputs. Introducing relative coordinates to Evoformer is crucial because it captures both position and rotation information between residues, providing substantially more information than pair-wise distances alone while keeping all structural inputs to the Evoformer SE(3)-invariant.

Structure module.

In the original structure module of AlphaFold2, a total of 8 weight-sharing layers transforms a zero-initialized conformation, $\{T_i = (\boldsymbol{I}, \mathbf{0})\}_{i=1}^n$, to a predicted conformation $\hat{\mathcal{P}}$. In Diffold, we reuse this module and its pre-trained parameters to iteratively refine $\mathcal{P}^{(t)}$ into $\hat{\mathcal{P}}_0^{(t)}$. The rationale behind this is that AlphaFold2 uses the learned weights to refine the zero-initialized conformation to the true structure, so we want to best utilize the pre-trained weights to refine noisy structures to multiple possible low-enery structures. Specifically, the diffused positions and rotations were properly scaled and fed to the module. After the alteration, the structural module retains its SE(3)-equivariance, yet a remaining question is whether the pre-trained weights can be smoothly adapted to diffused structures. Surprisingly, our early experiments show the per-trained weights of AlphaFold2 accomplish such robustness even without fine-tuning, as is demonstrated in B.4.

4.4 Hierarchical Reweighting of Training Samples

In AlphaFold2 and related protein folding models [14, 19, 39], the sequences of training samples are clustered, and reweighted to make all clusters have uniform weights. This approach has two major drawbacks. First, it is questionable whether uniformly sampling sequence clusters is optimal. For example, this protocol assigns the same weights to the family of antibodies with over 7,000 different sequences, and a rarely studied family with few sequences. This may also be one reason why AlphaFold2 performs unsatisfactorily on local precision of antibody structures [40]. Second, AlphaFold2 assigns the same sampling weights to conformations within the same sequence cluster, which introduces biases from the uneven data collection practices of the PDB: some conformations are well-studied with abundant structures, while others may have sparser data because they are newly discovered or less studied. Also for homomers, authors may display an assembly either with one chain and roto-translations, or multiple chains of identical conformations, which significantly affects the distribution.

To address the above issues, we developed a hierarchical sampling protocol for Diffold training. We first clustered unique sequences in PDB into sequence clusters $\mathcal{C}$s representing protein families. We then calculated pairwise TMscores within each cluster and constructed a weighted graph. The Louvain community detection method was then applied to each graph to sub-cluster it into fold clusters $\mathcal{F}$s. We finally assign sampling weights hierarchically to each sequence and fold cluster based on the logarithm of their sizes, and each conformation is then reweighted by its size following

$$w(\mathcal{P}) = k_s(|\mathcal{C}|) \times \frac{k_f(|\mathcal{F}|)}{\sum_{\mathcal{F}_i \subseteq \mathcal{C}} k_f(|\mathcal{F}_i|)} \times k_l(|\mathcal{P}|). \tag{16}$$

This reweighting protocol balances the distributions of protein families, conformational changes, and lengths in the training data. These balances are especially important for Diffold as a probabilistic generative model, which is expected to diversely sample conformations from the distribution derived from the training data. More details are provided in A.2.

4.5 Loss

The general form of loss in Diffold is defined as

$$L = \mathbb{E}_{\mathcal{P}_0 \in \mathcal{D}, t \in U(0,1), \mathcal{P}^{(t)} \sim p(\mathcal{P}, |\mathcal{P}_0, t)} \eta^t \mathcal{L}(\hat{\mathcal{P}}, \mathcal{P}_0; \mathcal{S}). \tag{17}$$

For the choice of $\mathcal{L}$, previous work [11] shows that using invariant metrics to measure distances between generated and target conformations is inappropriate, as the output conformation should be in the same global frame as the input for correct calculations of score functions. Although the original *frame aligned point error* (FAPE) loss proposed in AlphaFold2 has multiple benefits, it is invariant to roto-translations and thus should be modified. To utilize these benefits while avoiding

invariance issues, we add two auxiliary losses with small weights to prevent global shifts: 1) the errors of C_α coordinates between predicted and target conformations (without alignment), and 2) an L2 norm over the structure module's per-layer updates, which lie in the tangent space at the input structures per-layer. Notably the second loss provides twofold benefits: first, it encourages small changes from the input to avoid unnecessary shifts; second, it regularizes against overfitting and encourages iterative refinements. Other AlphaFold2 losses like pLDDT, distogram and violations are retained. Details are shown in A.3.

4.6 Langevin Dynamics and Structural Interpolation

One specific advantage of generating $\hat{\mathcal{P}}^{(0)}$ instead of score functions from the model is that this allows new approaches to using the model, such as the generation of continuous trajectories between conformations. We introduce two potential approaches to manipulate Diffold, namely the (overdamped) Langevin dynamics and structural interpolation.

Langevin dynamics.

The advantage of introducing Langevin dynamics is that this provides a way of recursive, continuous sampling, helpful in studying the dynamic nature of protein systems on larger time scales. Formally, overdamped Langevin dynamics are defined as

$$\mathrm{d}\boldsymbol{x} = -\frac{\epsilon}{k_B T}\nabla U(\boldsymbol{x})\mathrm{d}t + \sqrt{2\epsilon}\mathrm{d}\boldsymbol{w}^{\Omega_{\boldsymbol{x}}}, \tag{18}$$

which theoretically converges to a stationary, Boltzmann distribution $p(\boldsymbol{x}) = e^{-U(\boldsymbol{x})/k_B T}$. The process can be used to sample from $p(\mathcal{P}, t)$ with $U(\mathcal{P}) := -k_B T \log p(\mathcal{P}, t)$:

$$\mathrm{d}\mathcal{P} = \epsilon \nabla_{\mathcal{P}} \log p(\mathcal{P}, t)\mathrm{d}s + \sqrt{2\epsilon}\mathrm{d}\boldsymbol{w}^{\Omega_{\mathcal{P}}}, \tag{19}$$

where the scores can be approximated by diffusion models. Notably in Eq. (19), $s \in \mathbb{R}$ is the time of Langevin dynamics, while $t \in (0, 1]$ is the (fixed) time of diffusion. Prior work [41] has explored simulating dynamics with learned scores from diffusion models at $t \to 0$, but was limited to specific proteins and required massive MD simulations for training. We observed that this framework can be extended to any t by revealing the connection between Langevin dynamics and diffusion processes: the Langevin equation is equivalent to moving infinitesimally in the forward diffusion, and then in the reverse direction. This can be immediately proved by observing that Eq. (19) is a direct result of summing Eq. (3) and (4) with $\epsilon = g(t)$. More discussions and details are included in B.5.

Structural interpolation.

Another typical application of protein dynamics studies is to find a continuous trajectory of changes from one conformation to another. In computer vision, interpolations of diffused states are commonly used to fuse two images [31]. In this work, we extend the idea of image interpolation to structural interpolation. For two given conformations $\mathcal{P}_{0,A}$ and $\mathcal{P}_{0,B}$, the structural interpolation between two structures is done by first diffusing the two backbones to certain time t as $(T_A^{(t)}, T_B^{(t)}) = (c_A^{(t)}, \boldsymbol{R}_A^{(t)}; c_B^{(t)}, \boldsymbol{R}_B^{(t)})$, performing linear interpolation in-between as shown in Eq. (20) and Eq. (21), and then integrating the reverse-time diffusion.

$$\boldsymbol{c}^{(t)}(\lambda) = (1-\lambda)\boldsymbol{c}_A^{(t)} + \lambda \boldsymbol{c}_B^{(t)}, \quad \lambda \in [0, 1]. \tag{20}$$

$$\boldsymbol{R}^{(t)}(\lambda) = \boldsymbol{R}_A^{(t)} \cdot \mathrm{Exp}\left(\lambda \mathrm{Log}\left(\boldsymbol{R}_B^{(t)}(\boldsymbol{R}_A^{(t)})^\top\right)\right), \quad \lambda \in [0, 1], \tag{21}$$

References

[1] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., *et al.*: Highly accurate protein structure prediction with alphafold. Nature **596**(7873), 583–589 (2021)

[2] Oleinikovas, V., Saladino, G., Cossins, B.P., Gervasio, F.L.: Understanding cryptic pocket formation in protein targets by enhanced sampling simulations. Journal of the American Chemical Society **138**(43), 14257–14263 (2016)

[3] Tuckerman, M.E.: Statistical mechanics: theory and molecular simulation (2023)

[4] Sugita, Y., Okamoto, Y.: Replica-exchange molecular dynamics method for protein folding. Chemical physics letters **314**(1-2), 141–151 (1999)

[5] Barducci, A., Bonomi, M., Parrinello, M.: Metadynamics. Wiley Interdisciplinary Reviews: Computational Molecular Science **1**(5), 826–843 (2011)

[6] Kästner, J.: Umbrella sampling. Wiley Interdisciplinary Reviews: Computational Molecular Science **1**(6), 932–942 (2011)

[7] Del Alamo, D., Sala, D., Mchaourab, H.S., Meiler, J.: Sampling alternative conformational states of transporters and receptors with alphafold2. Elife **11**, 75751 (2022)

[8] Wayment-Steele, H.K., Ojoawo, A., Otten, R., Apitz, J.M., Pitsawong, W., Hömberger, M., Ovchinnikov, S., Colwell, L., Kern, D.: Predicting multiple conformations via sequence clustering and alphafold2. Nature, 1–3 (2023)

[9] Zheng, S., He, J., Liu, C., Shi, Y., Lu, Z., Feng, W., Ju, F., Wang, J., Zhu, J., Min, Y., et al.: Towards predicting equilibrium distributions for molecular systems with deep learning. arXiv preprint arXiv:2306.05445 (2023)

[10] Jing, B., Berger, B., Jaakkola, T.: Alphafold meets flow matching for generating protein ensembles. In: NeurIPS 2023 AI for Science Workshop (2023)

[11] Watson, J.L., Juergens, D., Bennett, N.R., Trippe, B.L., Yim, J., Eisenach, H.E., Ahern, W., Borst, A.J., Ragotte, R.J., Milles, L.F., *et al.*: De novo design of protein structure and function with rfdiffusion. Nature **620**(7976), 1089–1100 (2023)

[12] Yim, J., Trippe, B.L., De Bortoli, V., Mathieu, E., Doucet, A., Barzilay, R., Jaakkola, T.: Se (3) diffusion model with application to protein backbone generation. arXiv preprint arXiv:2302.02277 (2023)

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

[14] Li, Z., Liu, X., Chen, W., Shen, F., Bi, H., Ke, G., Zhang, L.: Uni-fold: an open-source platform for developing protein folding models beyond alphafold. bioRxiv, 2022–08 (2022)

[15] Steinegger, M., Söding, J.: Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. Nature biotechnology **35**(11), 1026–1028 (2017)

[16] Mirdita, M., Schütze, K., Moriwaki, Y., Heo, L., Ovchinnikov, S., Steinegger, M.: Colabfold: making protein folding accessible to all. Nature methods **19**(6), 679–682 (2022)

[17] Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.-Y.: Do transformers really perform badly for graph representation? Advances in neural information processing systems **34**, 28877–28888 (2021)

[18] Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., Tang, J.: Geodiff: A geometric diffusion model for molecular conformation generation. arXiv preprint arXiv:2203.02923 (2022)

[19] Ahdritz, G., Bouatta, N., Kadyan, S., Xia, Q., Gerecke, W., O'Donnell, T.J., Berenberg, D., Fisk, I., Zanichelli, N., Zhang, B., et al.: Openfold: Retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization. bioRxiv, 2022–11 (2022)

[20] Boeszoermenyi, A., Bernaleau, L., Chen, X., Kartnig, F., Xie, M., Zhang, H., Zhang, S., Delacrétaz, M., Koren, A., Hopp, A.-K., *et al.*: A conformation-locking inhibitor of slc15a4 with tasl proteostatic anti-inflammatory activity. Nature Communications **14**(1), 6626 (2023)

[21] Hassan, N.M., Alhossary, A.A., Mu, Y., Kwoh, C.-K.: Protein-ligand blind docking using quickvina-w with inter-process spatio-temporal integration. Scientific reports **7**(1), 15451 (2017)

[22] Koes, D.R., Baumgartner, M.P., Camacho, C.J.: Lessons learned in empirical scoring with smina from the csar 2011 benchmarking exercise. Journal of chemical information and modeling **53**(8), 1893–1904 (2013)

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

[24] Le Guilloux, V., Schmidtke, P., Tuffery, P.: Fpocket: an open source platform for ligand pocket detection. BMC bioinformatics **10**(1), 1–11 (2009)

[25] Zhou, G., Gao, Z., Ding, Q., Zheng, H., Xu, H., Wei, Z., Zhang, L., Ke, G.: Uni-Mol: A universal 3d molecular representation learning framework. In: The Eleventh International Conference on Learning Representations (2023). https://openreview.net/forum?id=6K2RM6wVqKu

[26] Alcaide, E., Li, Z., Zheng, H., Gao, Z., Ke, G.: Umd-fit: Generating realistic ligand conformations for distance-based deep docking models. In: NeurIPS 2023 Generative AI and Biology (GenBio) Workshop (2023)

[27] Müller, C., Schlauderer, G., Reinstein, J., Schulz, G.E.: Adenylate kinase motions during catalysis: an energetic counterweight balancing substrate binding. Structure **4**(2), 147–156 (1996)

[28] Evans, R., O'Neill, M., Pritzel, A., Antropova, N., Senior, A., Green, T., Žídek, A., Bates, R., Blackwell, S., Yim, J., et al.: Protein complex prediction with alphafold-multimer. biorxiv, 2021–10 (2021)

[29] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: International Conference on Machine Learning, pp. 2256–2265 (2015). PMLR

[30] Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)

[31] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456 (2020)

[32] Luo, S., Su, Y., Peng, X., Wang, S., Peng, J., Ma, J.: Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. Advances in Neural Information Processing Systems **35**, 9754–9767 (2022)

[33] Ingraham, J.B., Baranov, M., Costello, Z., Barber, K.W., Wang, W., Ismail, A., Frappier, V., Lord, D.M., Ng-Thow-Hing, C., Van Vlack, E.R., et al.: Illuminating protein space with a programmable generative model. Nature, 1–9 (2023)

[34] Jing, B., Erives, E., Pao-Huang, P., Corso, G., Berger, B., Jaakkola, T.: EigenFold: Generative protein structure prediction with diffusion models. arXiv preprint arXiv:2304.02198 (2023)

[35] De Bortoli, V., Mathieu, E., Hutchinson, M., Thornton, J., Teh, Y.W., Doucet, A.: Riemannian score-based generative modelling. Advances in Neural Information Processing Systems **35**, 2406–2422 (2022)

[36] Nikolayev, D.I., Savyolov, T.I., *et al.*: Normal distribution on the rotation group so (3). Texture, Stress, and Microstructure **29**, 201–233 (1997)

[37] Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 8162–8171. PMLR, ??? (2021)

[38] He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016, pp. 630–645. Springer, Cham (2016)

[39] Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., *et al.*: Evolutionary-scale prediction of atomic-level protein structure with a language model. Science **379**(6637), 1123–1130 (2023)

[40] Polonsky, K., Pupko, T., Freund, N.T.: Evaluation of the Ability of AlphaFold to Predict the Three-Dimensional Structures of Antibodies and Epitopes. The Journal of Immunology **211**(10), 1578–1588 (2023) https://doi.org/10.4049/jimmunol.2300150 https://journals.aai.org/jimmunol/article-pdf/211/10/1578/1646455/ji2300150.pdf

[41] Arts, M., Garcia Satorras, V., Huang, C.-W., Zügner, D., Federici, M., Clementi, C., Noé, F., Pinsler, R., Berg, R.: Two for one: Diffusion models and force fields for coarse-grained molecular dynamics. Journal of Chemical Theory and Computation **19**(18), 6151–6159 (2023)

[42] Zhang, Y., Skolnick, J.: Tm-align: a protein structure alignment algorithm based on the tm-score. Nucleic acids research **33**(7), 2302–2309 (2005)

[43] Kempen, M., Kim, S.S., Tumescheit, C., Mirdita, M., Gilchrist, C.L., Söding, J., Steinegger, M.: Foldseek: fast and accurate protein structure search. Biorxiv, 2022–02 (2022)

[44] Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008**(10), 10008 (2008)

[45] DeLano, W.L., *et al.*: Pymol: An open-source molecular graphics tool. CCP4 Newsl. Protein Crystallogr **40**(1), 82–92 (2002)

Acknowledgments

The computational resources were supported by the computing platform of DP technology and high-performance computing platform of Peking University.

Author information

Authors and Affiliations

School of Physics, Peking University, Beijing, China
Jiahao Fan & Huaqing Huang

Center for data scienc, Peking University, Beijing, China
Ziyao Li

DP technology, Beijing, China
Jiahao Fan & Ziyao Li & Eric Alcaide & Guolin Ke

University of Barcelona, Barcelona, Spain
Eric Alcaide

School of Mathematical Sciences, Peking University, Beijing,China
Weinan E

Corresponding authors

Correspondence to Guolin Ke, Huaqing Huang or Weinan E.

Ethics declarations

Competing interests

The authors declare no competing interests.

A Implementation Details

A.1 Markovian Span-Masking and Missing Residues

To recap, the span-masking strategy is defined as:

$$M_i \sim \text{Bernoulli}(p_i),\ i \geq 1, \qquad p_1 = \rho,$$
$$p_i = \frac{K-1}{K} M_{i-1} + \frac{\rho}{(1-\rho)K}(1 - M_{i-1}),\ i > 1,$$

where $M_i = 1$ indicates residue i is selected as a motif and not diffused during the forward process. Notably, when i is a residue without experimentally solved structure, M_i is always set as 0.

We now prove that 1) the overall masking rate is ρ, i.e., $\mathbb{E}(\frac{1}{n}\sum_{i=1}^{n} M_i) = \rho$ and 2) the expected number of consecutive motifs is K. To show 1), by induction it can be easily verified that $P(M_i = 1) = \rho$. Therefore,

$$\mathbb{E}\left(\frac{1}{n}\sum_{i=1}^{n} M_i\right) = \frac{1}{n}\sum_{i=1}^{n} P(M_i = 1) = \rho. \tag{S22}$$

For 2), note that $P(M_{i+1} = 1 | M_i = 1) = 1 - 1/K$. Therefore, conditioned on $M_i = 1$, the length of the sequence of consecutive 1s $(i, i+1, \cdots)$ follows a Geometric$(\frac{1}{K})$ distribution, and the expectation can be calculated as K.

In the training of Diffold, we set $K = 15$ and calculated ρ per sample as

$$\rho = \max(0, \tilde{\rho} - (1 - r)), \quad \tilde{\rho} \sim \text{Uniform}(0, 1), \quad r = 0.6. \tag{S23}$$

Additionally, no extra symbols were introduced to encode motifs. Instead, we assigned $t_i = 0$ to fully leverage the time embedding to represent that the residues have zero noise. We took a similar approach for missing residues without solved structures, drawing their structures from the prior and assigning $t_i = 1$.

A.2 Hierarchical Reweighting Details

We collected all protein chains that were released before April 30, 2022, and filtered out those without valid sequences. A total of 645,947 protein chains, with 133,954 unique sequences were collected. To cluster these unique sequences, we used MMseqs2 [15] with parameters `--min-seq-id 0.4 -c 0.8 --cov-mode 1`, which created sequence clusters sharing $> 40\%$ sequence identities. This generated 37,838 sequence clusters.

For the 22,122 sequence clusters with at least 3 sequences, we further divided them into fold clusters by graph partition. We used TMalign [42] to calculate all pairwise TMscores inside each sequence cluster based on the implementation of Foldseek [43]. Notably, this implementation prefilters and avoids calculating TMscores for pairs unlikely to have high similarity. We assigned a TMscore of 0 for such pairs. All calculated TMscores (r) were processed into link weights via the kernel

$$w = \left(\frac{\max(0, r - r_0)}{1 - r_0}\right)^m \tag{S24}$$

with $r_0 = 0.85$ and $m = 2$. Notably, this kernel ignores connections between structures with $r < 0.85$. We partitioned the generated graph using the Louvain algorithm [44] to produce communities, i.e., the fold clusters. This generated 41,567 fold clusters.

To recap, the final sampling weight $w(\mathcal{P})$ of each conformation $\mathcal{P} \in \mathcal{F}, \mathcal{F} \subseteq \mathcal{C}$ was calculated as:

$$w(\mathcal{P}) = k_s(|\mathcal{C}|) \times \frac{k_f(|\mathcal{F}|)}{\sum_{\mathcal{F}_i \subseteq \mathcal{C}} k_f(|\mathcal{F}_i|)} \times \frac{1}{|\mathcal{F}|} \times k_l(|\mathcal{P}|).$$

Specifically, we used:

$$k_s(x) = \log_{10}(x + 9) \tag{S25}$$
$$k_f(x) = \log_{10}(x + 99) \tag{S26}$$
$$k_l(x) = \frac{1}{2}\left(1 - \cos\left(\frac{\pi}{2} \cdot \max(512, \min(x, 32))\right)\right) \tag{S27}$$

Intuitively, the sampling probability of each cluster is still relative to its size, but much smoothed by logarithm. The bias to k_f is larger than k_s because we want the conformations to be sampled more evenly (weights ranging from 2.0 to 3.5).

Sequences with very short lengths are punished by k_l, keeping a minimal sampling weight ≈ 0.01. Fig. S6 shows more details on the distributions and impacts of so-defined sampling protocol.

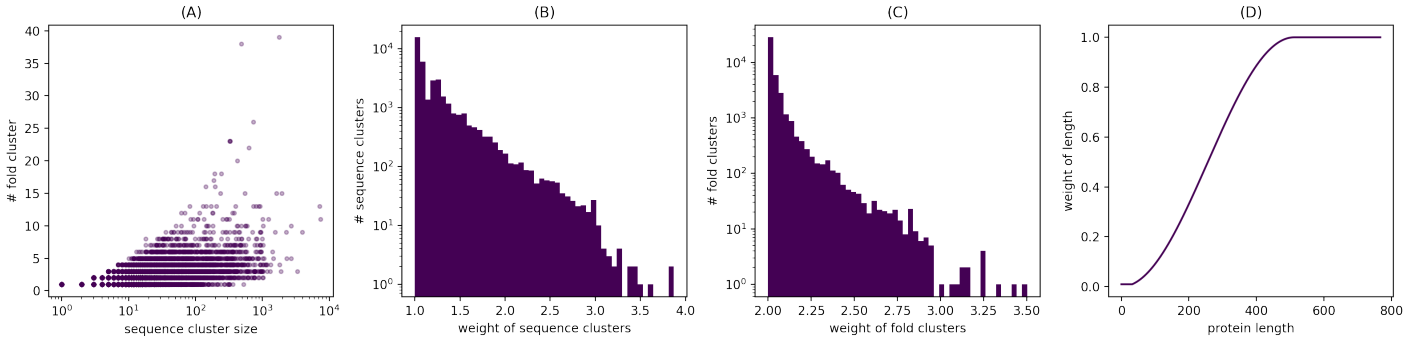


Fig. S6: Details of the hierarchical reweighting protocol. (A) number of fold clusters versus sequence cluster sizes. (B) distribution of sequence cluster weights; (C) distribution of fold cluster weights; (D) length weight versus protein length.

A.3 Loss

As is discussed in Section 4.5, we use the original losses introduced in AlphaFold as well as 2 additional losses, namely a loss of C_α coordinate errors and a penalty over the update vector. The formal definition of the loss we use is

$$\mathcal{L} = \mathcal{L}_{\text{FAPE}} + \mathcal{L}_{\text{chi}} + 0.1\mathcal{L}_{\text{penalty}} + 0.1\mathcal{L}_{\text{ca-error}} + 0.02\mathcal{L}_{\text{violation}} + \mathcal{L}_{\text{aux}}, \tag{S28}$$

$$\mathcal{L}_{\text{aux}} = 2.0\mathcal{L}_{\text{BERT-MSA}} + 0.3\mathcal{L}_{\text{distogram}} + 0.01\mathcal{L}_{\text{plddt}} + 0.1\mathcal{L}_{\text{pae}} + 0.01\mathcal{L}_{\text{resolved}}, \tag{S29}$$

in which we inherited most of the loss weights in AlphaFold's pre-training. The definition of $\mathcal{L}_{\text{penalty}}$ and $\mathcal{L}_{\text{ca-error}}$ are

$$\mathcal{L}_{\text{ca-error}} = \frac{1}{n}\sum_{i=1}^{n} \|\hat{\boldsymbol{c}}_i - \boldsymbol{c}_i\|_2 \tag{S30}$$

$$\mathcal{L}_{\text{penalty}} = \frac{1}{n}\sum_{i=1}^{n}\sum_{l=1}^{L}\left(\|\hat{\boldsymbol{c}}_i^{(l)} - \hat{\boldsymbol{c}}_i^{(l-1)}\|_2 + \|\boldsymbol{q}_i^{(l)}\|_2\right), \qquad \hat{\boldsymbol{c}}_i^{(l)} = \boldsymbol{c}^{(t)} \tag{S31}$$

where $\boldsymbol{q}_i \in \mathbb{R}^3$ is the vector of (unnormalized) imaginary components in the quaternion representation of the update rotation. Notably, $\mathcal{L}_{\text{FAPE}}, \mathcal{L}_{\text{chi}}, \mathcal{L}_{\text{penalty}}$ and $\mathcal{L}_{\text{ca-error}}$ are calculated for per-layer outputs of the structure module. The FAPE loss of both backbones and sidechains are all calculated and weighted as $\mathcal{L}_{\text{FAPE}} = 0.5(\mathcal{L}_{\text{bb}} + \mathcal{L}_{\text{sc}})$.

A.4 Training and Inference Algorithm

We formulate the training algorithm of Diffold as Algorithm 1. For inference, we first introduce the discretization of the continuous form of the reverse-time SDE. We implemented the specific discretization form as

$$\boldsymbol{c}^{(t-\tau)} = \frac{1}{\sqrt{1-\beta_{\text{pos}}^{(t,t-\tau)}}}\left(\boldsymbol{c}^{(t)} + \beta_{\text{pos}}^{(t,t-\tau)}\nabla_{\boldsymbol{c}^{(t)}}\log p(\boldsymbol{c}^{(t)}, t|\hat{\boldsymbol{c}}^{(t)})\right) + \frac{1-\gamma_{\text{pos}}^{(t-\tau)}}{(1-\gamma_{\text{pos}}^{(t)})\beta_{\text{pos}}^{(t,t-\tau)}}\boldsymbol{\epsilon}_{\text{pos}}, \quad \boldsymbol{\epsilon}_{\text{pos}} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_3) \tag{S32}$$

$$\boldsymbol{R}^{(t-\tau)} = \text{Exp}\left(\beta_{\text{rot}}^{(t,t-\tau)}\nabla_{\boldsymbol{R}^{(t)}}\log p(\boldsymbol{R}^{(t)}, t|\hat{\boldsymbol{R}}^{(t)}) + \boldsymbol{\epsilon}_{\text{rot}}^{(t)}\right)\cdot\boldsymbol{R}^{(t)}, \quad \boldsymbol{\epsilon}_{\text{rot}}^{(t)} \sim \text{IGSO3}\left(\beta_{\text{rot}}^{(t,t-\tau)}\right). \tag{S33}$$

Specifically, Eq. (S32) is a reparameterization of the classic DDPM [30] instead of VP-SDE [31], because we find that the discretization form of VP-SDE suffers from numerical issues when $\beta \gg 0$. When drawing $\boldsymbol{\epsilon}_{\text{rot}} \sim \text{IGSO3}(\beta)$, we sample uniformly distributed rotation axis from the unit ball of $\mathbb{R}^3$ and sample a rotation angle from $\psi(\cdot, \beta)$.The parameters of $\beta_*^{(t,t-\tau)}$ are calculated as

$$\beta_{\text{pos}}^{(t,t-\tau)} = 1 - \frac{\gamma_{\text{pos}}^{(t)}}{\gamma_{\text{pos}}^{(t-\tau)}}, \tag{S34}$$

$$\beta_{\text{rot}}^{(t,t-\tau)} = \sigma_{\text{rot}}^{(t)} - \sigma_{\text{rot}}^{(t-\tau)}, \tag{S35}$$

which corresponds to the definitions of γ and σs. The detailed algorithm of inference is presented in Algorithm 3.

Algorithm 1 Diffold training.

Input: dataset $\mathcal{D} = \{(\mathcal{S}, \mathcal{P})\}$, pre-trained AlphaFold weights θ
Initialize model weights as θ.
repeat
  Sample $(\mathcal{S}, \mathcal{P})$ from $\mathcal{D}$ according to Eq. (16).
  Sample $t \sim \text{Uniform}(0, 1)$.
  Sample motif mask $\mathcal{M}$ according to Eq. (15).
  $T^{(t)} = \text{ForwardDiffusion}(t, \mathcal{P}, \mathcal{M})$ (see Algorithm 2).
  $\hat{\mathcal{P}} = m_\theta(t, T^{(t)}, \mathcal{S})$.
  Calculate $\mathcal{L}(\hat{\mathcal{P}}, \mathcal{P}; \mathcal{S})$ following Eq. (S28).
  Backpropagate $\mathcal{L}$ to update θ.
until convergence

Algorithm 2 ForwardDiffusion

Input: time $t \in (0, 1]$, protein $\mathcal{P} = (\mathcal{A}_i = ((\boldsymbol{R}_i, \boldsymbol{c}_i), \mathcal{X}_i))_{i=1}^n$, motif mask $\mathcal{M} = (M_1, \cdots, M_n)$
for $i = 1$ **to** n **do**
  if $M_i = 0$ (residue not fixed as motif) **then**
    if structure of $\mathcal{A}_i$ is missing **then**
      Draw $T_i^{(t)} = (\boldsymbol{R}_i^{(t)}, \boldsymbol{c}_i^{(t)})$ from prior.
    else
      Sample $T_i^{(t)} = ((\boldsymbol{R}_i^{(t)}, \boldsymbol{c}_i^{(t)}))$ following Eq. (7) (8) providing $\boldsymbol{R}_i, \boldsymbol{c}_i$.
    end if
  else
    $T_i^{(t)} = T_i = (\boldsymbol{R}_i, \boldsymbol{c}_i)$
  end if
end for
Return: $T^{(t)} = \left\{T_i^{(t)}\right\}_{i=1}^n$

Algorithm 3 Diffold inference.

Input: protein sequence $\mathcal{S}$, motif mask $\mathcal{M}$, motif structures $\{T_i^{\text{M}} | M_i = 1\}$, number of steps S
$t = 1, \tau = 1/S$.
Initialize $T^{(1)}$ given $\mathcal{M}, \{T_i^{\text{M}} | M_i = 1\}$ (see Algorithm 4).
for $s = 1$ **to** S **do**
  Calculate $\hat{\mathcal{P}} = m_\theta(t, T^{(t)}, \mathcal{S})$.
  $T^{(t-\tau)} = \text{ReverseDiffusion}(t, \tau, T^{(t)}, \hat{\mathcal{P}}, \mathcal{M})$ (see Algorithm 5).
  $t = t - \tau$
end for
Return $\mathcal{P}^{(0)}$

A.5 Training Details

The structural diffusion fine-tuning was initialized with parameters in Model 2 of the v3 parameters of AlphaFold-Multimer [1]. Table S3 summarizes the parameters used to train Diffold. Most of the settings are inherited from AlphaFold-Multimer (v3). The original parameters were adapted to Uni-Fold [14] implementation[2] using the provided utilities in the repository. Training data and generated MSAs were downloaded from the repository as well. Due to the cost of fine-tuning, we did not do a thorough hyperparameter search.

[1] https://storage.googleapis.com/alphafold/alphafold_params_2022-12-06.tar
[2] https://github.com/dptech-corp/Uni-Fold/

Algorithm 4 Initialize

Input: motif mask $\mathcal{M} = (M_1, \cdots, M_n)$, motif structures $\{\mathcal{A}_i^{\mathrm{M}} | M_i = 1\}$
for $i = 1$ **to** n **do**
  if $M_i = 0$ (not provided as motif) **then**
    Draw $T_i^{(t)} = (\boldsymbol{R}_i^{(t)}, \boldsymbol{c}_i^{(t)})$ from prior.
  else
    $T_i = T_i^{\mathrm{M}}$
  end if
end for
Return $T^{(1)} = (T_1, \cdots, T_n)$

Algorithm 5 ReverseDiffusion

Input: time $t \in (0, 1]$, interval $\tau \in (0, t]$, diffused protein backbone $T^{(t)}$, predicted protein $\hat{\mathcal{P}}$, motif mask $\mathcal{M}$.
if $t - \tau = 0$ **then**
  $\mathcal{P}^{(t-\tau)} = \hat{\mathcal{P}}$
else
  for $i = 1$ **to** n **do**
    if $M_i = 0$ (not provided as motif) **then**
      Calculate $T_i^{(t-\tau)}$ with Eq. (S32) - (S33) providing $T_i^{(t)}, \hat{T}_i, t, \tau$.
    else
      $T_i^{(t-\tau)} = T_i^{(t)}$
    end if
  end for
end if
Return $T^{(t-\tau)} = (T_1^{(t-\tau)}, \cdots, T_n^{(t-\tau)})$

Parameters	Values
number of samples	645,947
batch size	64
total steps	20,000
peak learning rate	1e-3
warm-up steps	1000
learning rate decay	N/A
number of MSAs	512
number of extra MSAs	2,048
sequence crop size	256
number of templates	N/A
dimension of MSA, pair and single representations	(256, 128, 384)
dimension of RBF of time	512
dimension of RBF of relative coordinates	64

Table S3: Details of parameters in Diffold.

B More Results

B.1 RAC-47 Curation Details

To curate RAC-47, we collected all 82777 protein chains in PDB [13] released between 2022-04-30 and 2024-03-29. MMseqs2 [15] was used to cluster the sequences with 100% sequence identity and clusters with sizes ranging from 2 to 10 were kept, resulting in 28409 chains. We then filtered for chains with 128 to 768 residues, yielding a set of 20078 chains. From the initially filtered 20078 protein structures, we identified 4256 sequence clusters, each containing at least one of these structures. Aligned backbone Root Mean Square Deviations (RMSDs) for all pairings within these clusters were calculated using PyMOL [45]. The primary selection criterion targeted clusters with at least one pair of RMSD greater than 2 Angstroms. This process resulted in 304 clusters of chains. But among these clusters, there exists clusters where the resolved structures have rather different sequences (due to the experimental details). For convenience we just select those clusters with the resolved structures having the same sequence, this procedure produces 55 clusters. Finally, we exclude the clusters where the resolved structures have incontinuous sequence, our pipeline yielded 47 unbiased protein clusters for

PDB IDs	Diff	Alphafold	Diffold	AlphaFlow	MSA-sub. (64)	MSA-sub. (256)	AF-cluster
7Z3N_C/7Z3O_C	0.658	0.567/0.651	0.634/0.571	0.558/0.673	0.556/0.638	0.566/0.634	0.498/0.598
7R5J_F0/7R5K_F0	0.461	0.388/0.387	0.395/0.392	0.418/0.428	0.389/0.407	0.395/0.390	0.354/0.392
7YCO_B/8H6F_X	0.543	0.966/0.538	0.968/0.556	0.971/0.547	0.973/0.550	0.970/0.539	0.955/0.576
8TOC_b/8TV9_a	0.658	0.411/0.402	0.392/0.396	0.386/0.418	0.433/0.424	0.408/0.408	N/A
8TH8_S/8TID_s	0.527	0.537/0.551	0.621/0.659	0.603/0.690	0.544/0.563	0.559/0.576	0.561/0.576
7P37_A/7P3F_A	0.657	0.726/0.808	0.784/0.823	0.727/0.821	0.753/0.706	0.731/0.778	0.731/0.725
7UTI_Z/7UTL_d	0.561	0.505/0.446	0.483/0.592	0.574/0.551	0.508/0.445	0.499/0.460	0.512/0.512
8BDV_A/8BH7_A	0.537	0.422/0.596	0.573/0.771	0.620/0.787	0.443/0.675	0.502/0.732	0.663/0.804
7SJO_H/7SJP_H	0.651	0.875/0.879	0.915/0.906	0.946/0.932	0.841/0.923	0.841/0.882	0.815/0.883
7TUC_A/7TUE_A	0.716	0.981/0.702	0.953/0.677	0.990/0.739	0.979/0.758	0.978/0.717	0.929/0.737
8DKF_H/8DOW_C	0.679	0.903/0.770	0.905/0.911	0.945/0.919	0.896/0.818	0.903/0.805	0.679/0.688
8EFR_A/8EFT_A	0.787	0.842/0.759	0.812/0.795	0.836/0.802	0.836/0.738	0.856/0.772	0.405/0.404
8E2Y_A/8E31_A	0.674	0.727/0.648	0.752/0.669	0.752/0.663	0.705/0.635	0.730/0.655	0.629/0.578
8D01_L/8DOY_L	0.698	0.876/0.870	0.950/0.919	0.912/0.876	0.967/0.723	0.930/0.863	0.835/0.780
8BFL_A/8BFP_A	0.739	0.892/0.860	0.929/0.911	0.855/0.895	0.874/0.878	0.880/0.910	0.271/0.280
7SJO_F/7SJP_L	0.699	0.925/0.956	0.942/0.983	0.956/0.830	0.880/0.973	0.923/0.970	0.621/0.743
8DUE_A/8DVF_A	0.720	0.775/0.790	0.822/0.944	0.882/0.934	0.844/0.801	0.810/0.807	0.449/0.459
7ZWM_E/7ZXF_E	0.712	0.937/0.958	0.841/0.980	0.925/0.984	0.830/0.976	0.947/0.977	0.623/0.707
7Y5A_D/7Y5B_D	0.829	0.893/0.935	0.943/0.961	0.945/0.948	0.916/0.940	0.894/0.939	0.847/0.923
8FWF_L/8FYM_L	0.749	0.957/0.919	0.981/0.896	0.985/0.978	0.983/0.949	0.974/0.925	0.722/0.623
8B6V_A/8B6W_A	0.655	0.964/0.669	0.973/0.707	0.956/0.750	0.950/0.758	0.969/0.706	0.362/0.359
8DKF_L/8DOW_D	0.764	0.943/0.955	0.981/0.962	0.972/0.957	0.979/0.943	0.984/0.957	0.916/0.889
7WKP_A/7WWU_I	0.723	0.921/0.695	0.950/0.730	0.926/0.724	0.929/0.705	0.937/0.699	0.925/0.689
7PIS_9/7PIT_9	0.872	0.721/0.716	0.868/0.916	0.885/0.909	0.878/0.921	0.871/0.911	0.290/0.297
8DKE_P/8DKI_P	0.813	0.963/0.963	0.949/0.941	0.953/0.960	0.963/0.945	0.947/0.968	0.960/0.855
8G4C_D/8G4D_D	0.808	0.856/0.841	0.916/0.883	0.879/0.847	0.845/0.787	0.864/0.829	0.672/0.630
7ZF5_D/7ZF6_H	0.801	0.956/0.937	0.924/0.944	0.972/0.901	0.928/0.933	0.910/0.950	0.676/0.620
8HKX_AS7P/8HKY_AS7P	0.755	0.797/0.771	0.803/0.796	0.800/0.788	0.782/0.748	0.790/0.763	0.782/0.755
8HFX_E/8IFY_E	0.890	0.915/0.888	0.927/0.903	0.916/0.886	0.918/0.893	0.921/0.893	0.905/0.877
8SQZ_C/8SRM_C	0.754	0.687/0.862	0.684/0.846	0.709/0.859	0.691/0.862	0.689/0.864	0.682/0.855
8HKX_S13P/8HKY_S13P	0.769	0.786/0.768	0.797/0.772	0.801/0.766	0.788/0.759	0.782/0.770	0.741/0.731
8HKY_AL1P/8HKZ_AL1P	0.826	0.794/0.770	0.816/0.768	0.825/0.784	0.793/0.776	0.800/0.777	0.795/0.776
8V2D_a/8V3B_A	0.874	0.826/0.860	0.817/0.813	0.944/0.907	0.835/0.863	0.829/0.861	N/A
8E39_A/8E3A_A	0.757	0.724/0.649	0.733/0.683	0.783/0.696	0.672/0.670	0.735/0.681	0.533/0.507
8TCA_L/8VEV_B	0.840	0.895/0.978	0.951/0.976	0.960/0.967	0.975/0.971	0.906/0.976	0.899/0.932
8EE5_L/8EF3_B	0.843	0.696/0.871	0.807/0.954	0.952/0.980	0.912/0.959	0.929/0.933	0.807/0.940
8EE5_H/8EF3_A	0.846	0.824/0.681	0.936/0.908	0.966/0.956	0.866/0.812	0.844/0.721	0.838/0.778
7ZF5_F/7ZF6_L	0.856	0.976/0.949	0.976/0.992	0.990/0.959	0.967/0.980	0.989/0.970	0.665/0.701
8HC3_E/8HC6_L	0.853	0.898/0.889	0.916/0.917	0.934/0.932	0.898/0.905	0.856/0.827	0.893/0.865
7QD7_AAA/7QH9_AAA	0.900	0.985/0.957	0.985/0.981	0.984/0.973	0.977/0.960	0.983/0.954	0.305/0.306
7T22_A/7T23_B	0.707	0.849/0.707	0.931/0.870	0.879/0.753	0.821/0.811	0.836/0.704	0.804/0.722
8UOP_A/8UOY_B	0.898	0.907/0.936	0.918/0.943	0.918/0.941	0.921/0.936	0.912/0.936	0.907/0.934
7S8G_L/7UDS_L	0.856	0.866/0.923	0.935/0.947	0.975/0.962	0.736/0.676	0.773/0.700	0.827/0.895
8I6O_B/8I6Q_B	0.862	0.895/0.916	0.943/0.925	0.928/0.920	0.908/0.918	0.908/0.915	0.915/0.916
8GAE_E/8GFT_E	0.898	0.674/0.675	0.676/0.678	0.676/0.675	0.674/0.675	0.674/0.675	0.672/0.675
8HC3_H/8HC5_H	0.878	0.868/0.840	0.909/0.912	0.875/0.894	0.819/0.775	0.856/0.865	0.634/0.668
8JSG_A/8JSH_A	0.849	0.816/0.730	0.820/0.762	0.828/0.761	0.863/0.809	0.843/0.773	0.808/0.798
Mean-of-Best	0.753	0.811/0.783	0.839/0.824	0.850/0.826	0.818/0.793	0.822/0.794	0.696/0.688
Median-of-Best	0.757	0.866/0.808	0.915/0.896	0.916/0.876	0.863/0.809	0.856/0.807	0.722/0.722

Table S4: Evaluation results on RAC-47. *Diff.* indicates the TM-score between two real conformations. The best TM-score of 100 generated conformations to both of the real ones in each case are reported and aggregated. MSA-sub. represents MSA-subsampling method.

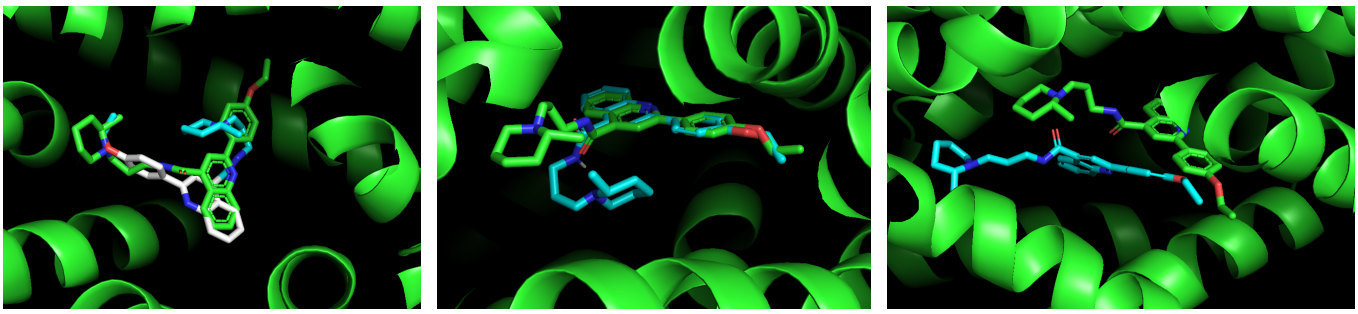
analysis. To evaluate the sampling accuracy and diversity in a convenient way, two protein chains with the largest RMSD in each cluster are selected as reference experimental structures.

B.2 Detailed results on RAC-47

In Table S4 we show the best TM-score among 100 generated conformations corresponding to two real conformations for each case in our **RAC-47** dataset. Note that some cases are neglected for AF-cluster since there are no clusters generated using the default settings in AF-cluster.

B.3 Results of existing docking methods in redocking setup

The results of three docking methods are shown in Figure Fig. S7, the binding ligand RMSD with the experimental structure is 5.997 Å for QuickVina-W, 6.633 Å for Smina and 8.543 Å for Diffdock. The results indicate that the three docking methods fail to dock the ligand to the right binding mode using the experimental structure (8JZX).



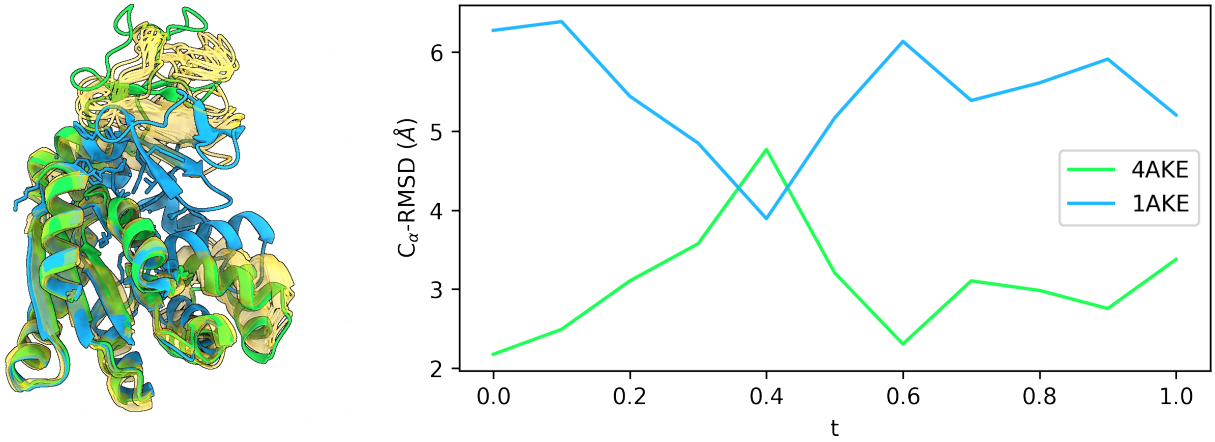
(a) QuickVina-W (b) Smina (c) Diffdock

Fig. S7: Binding sites from three docking methods: (a) QuickVina-W (b) Smina (c) Diffdock. Green: Experimental strucuture of 8JZX and its binding ligand; blue: Docked ligand structure for different docking methods.

B.4 Structure Module on Diffused Inputs

In our early experiments, we tested the performance of AlphaFold with changed inputs to the structure module before fine-tuning. Surprisingly, replacing the zero-initialized structures with diffused structures did not affect the model's overall performance. We posit that the weight-sharing mechanism contributes to this robustness, as it was previously used to refine all zero-initialized and half-generated structures.

Specifically, we diffused the conformation of 4AKE with different diffusion times $t = (0.0, 0.1, \cdots, 1.0)$, and inputted these diffused structures into the structure module to generate predictions. From the metrics and visualized results reported in Fig. S8, the structure module generates conformations from diffused inputs with good qualities. Interestingly, the generated conformations are in-between the apo and holo states of the protein. Even when the un-diffused structure was inputted ($t = 0$), the model bent it slightly towards the conformation of 1AKE.



(a) Generated conformations. (b) RMSD to 4AKE and 1AKE.

Fig. S8: AlphaFold behaves surprisingly well on diffused structures even without fine-tuning. In (a), green: 4AKE; blue: 1AKE; yellow: generated conformations from AlphaFold's structure module with diffused 4AKE with $t = (0.0, 0.1, \cdots, 1.0)$.

B.5 Details on Langevin Dynamics

We hereby specify the implementation of the overdamped Langevin dynamics. Specifically, we implement each step of the overdamped Langevin by taking a step in the reverse-time SDE (Eq. (4)) and then a step (with the same size) in the forward SDE (Eq. (3)). Specifically, the reverse-time SDE was discretized following Eq. (S32) - (S33), and the discretization of the forward SDE was introduced in a corresponding form as

$$\boldsymbol{c}^{(t)} = \sqrt{1 - \beta_{\text{pos}}^{(t,t-\tau)}}\boldsymbol{c}^{(t-\tau)} + \beta_{\text{pos}}^{(t,t-\tau)}\boldsymbol{\epsilon}_{\text{pos}}, \quad \boldsymbol{\epsilon}_{\text{pos}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_3) \tag{S36}$$

$$\boldsymbol{R}^{(t)} = \text{Exp}\left(\boldsymbol{\epsilon}_{\text{rot}}^{(t-\tau)}\right) \cdot \boldsymbol{R}^{(t-\tau)}, \quad \boldsymbol{\epsilon}_{\text{rot}}^{(t-\tau)} \sim \text{IGSO3}\left(\beta_{\text{rot}}^{(t,t-\tau)}\right). \tag{S37}$$

The implementation can be justified in two ways: 1) by observing that Eq. (19) is formally equivalent to the sum of Eq. (4) and Eq. (3); 2) by observing that this implementation does not change the marginal distributions of $\mathcal{P}$, therefore providing a sampling method of $p(\mathcal{P}, t)$.

Notably, when analyzing the generated trajectories, we are more focused on the transition and distribution of the per-step predictions $\hat{\mathcal{P}}_0^{(t)}$, which have correct geometries and *conform to a distribution induced by the model* m_θ *from* $p(\mathcal{P}^{(t)}, t)$. Under condition $t \to 0$, as is required in [41], the two spaces tend to be equivalent. However, drastic changes of conformation can hardly be observed when $t \to 0$, because the diffused structure $\mathcal{P}^{(t)}$ contains too much information and the model would be trapped into the local minimum around $\mathcal{P}^{(t)}$.

Besides Fig. 5 (e-f), we also plot the 2D projection of trajectories in terms of RMSD to two conformations as Fig. S9. Notably, the trajectory well covers a bridge between 1AKE and 4AKE with continuity.

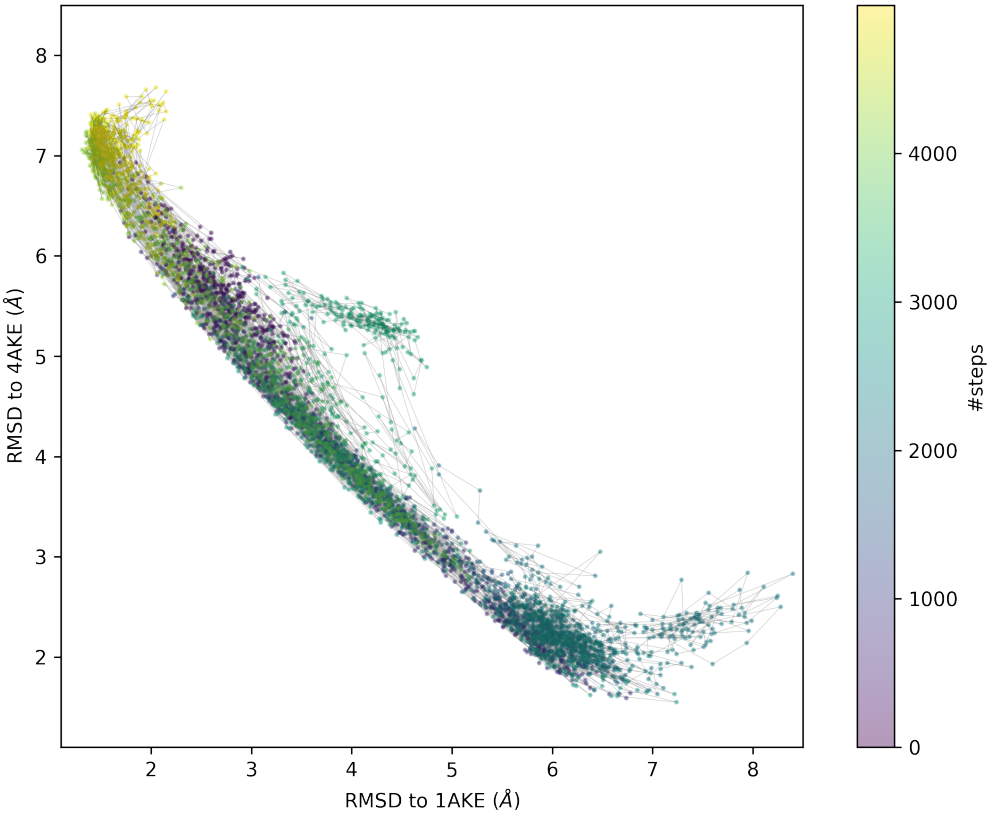


Fig. S9: Trajectory of 5000 steps of Langevin dynamics

.

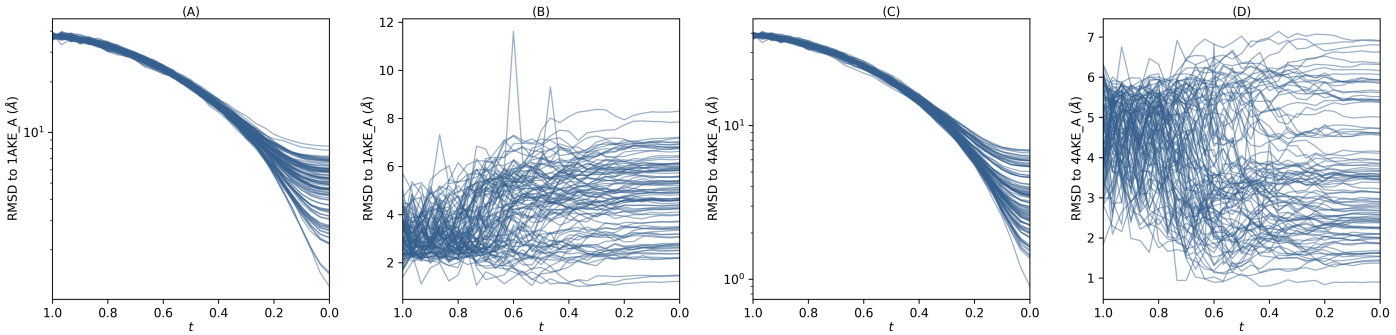
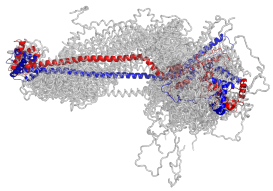


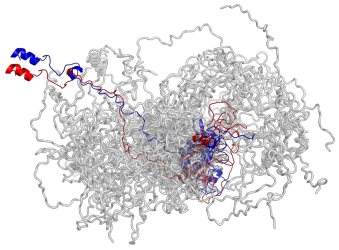
Fig. S10: Per-step RMSDs to 1AKE / 4AKE in the inference process.(A) RMSDs of noisy structures versus 1AKE; (B) RMSDs of predicted structures versus 1AKE; (C) RMSDs of noisy structures versus 4AKE; (D) RMSDs of predicted structures versus 4AKE.

B.6 More Visualization Results

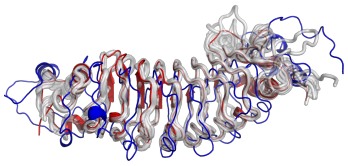
We visualize the inference process on 1AKE case by Diffold, as shown in Fig. S10. A balanced sampling between 1AKE and 4AKE is observed. The diffused conformations, $\mathcal{P}^{(t)}$, converge gradually to conformations with correct geometries. The predicted conformations, $\hat{\mathcal{P}}^{(t)}$, retain correct geometries all the time and are all close to real conformations. Global folding is determined during the early stage of inference, while the local precision is enhanced (not embodied in the figure) during the late half. For all conformations generated for RAC-47 from Diffold, we visualize them in Fig. S11-Fig. S14.



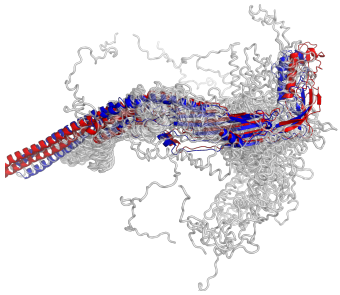
(a) 7Z3N_C/7Z3O_C



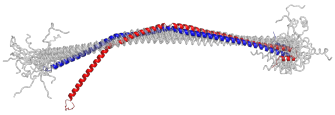
(b) 7R5J_F0/7R5K_F0



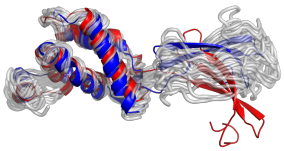
(c) 7YCO_B/8H6F_X



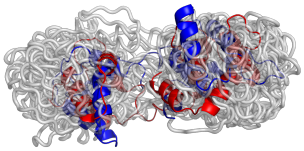
(d) 8TOC_b/8TV9_a



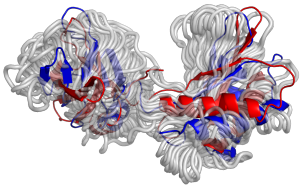
(e) 8TH8_S/8TID_s



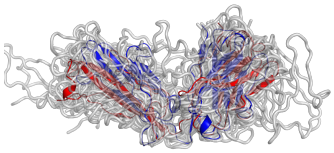
(f) 7P37_A/7P3F_A



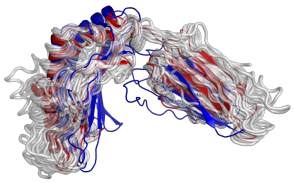
(g) 7UTI_Z/7UTL_d



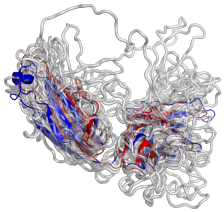
(h) 8BDV_A/8BH7_A



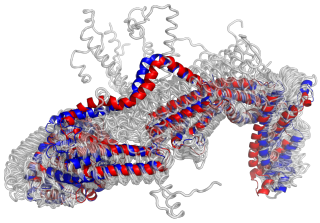
(i) 7SJO_H/7SJP_H



(j) 7TUC_A/7TUE_A

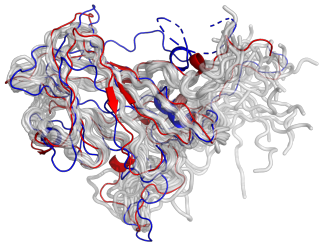


(k) 8DKF_H/8DOW_C

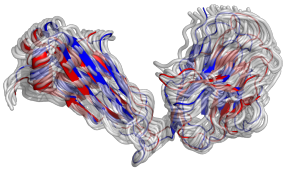


(l) 8EFR_A/8EFT_A

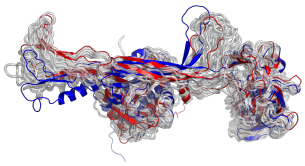
Fig. S11: Comparative visualization of PDB in **RAC-47** benchmark (Part 1)



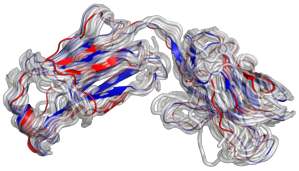
(a) 8E2Y_A/8E31_A



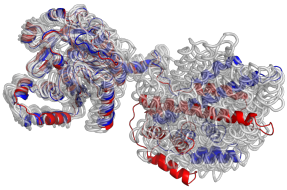
(b) 8D01_L/8D0Y_L



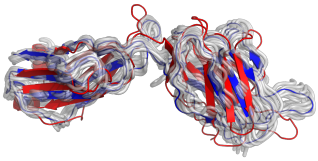
(c) 8BFL_A/8BFP_A



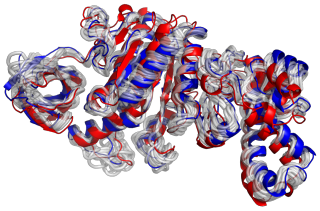
(d) 7SJO_F/7SJP_L



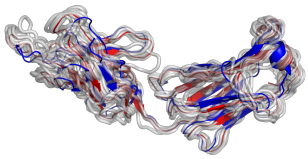
(e) 8DUE_A/8DVF_A



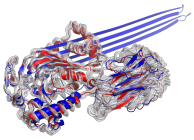
(f) 7ZWM_E/7ZXF_E



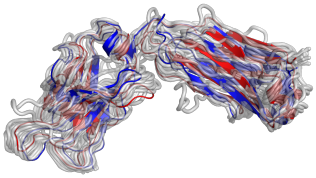
(g) 7Y5A_D/7Y5B_D



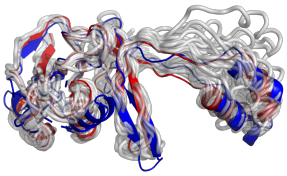
(h) 8FWF_L/8FYM_L



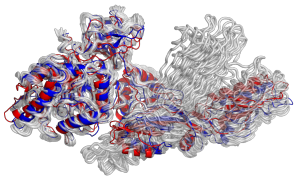
(i) 8B6V_A/8B6W_A



(j) 8DKF_L/8DOW_D

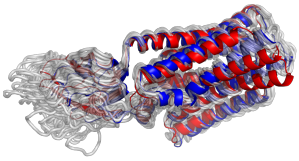


(k) 7WKP_A/7WWU_I

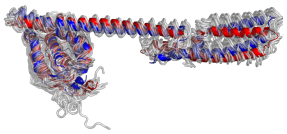


(l) 7PIS_9/7PIT_9

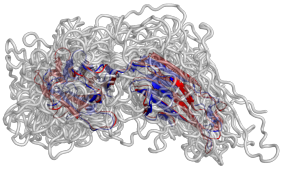
Fig. S12: Comparative visualization of PDB in **RAC-47** benchmark (Part 2)



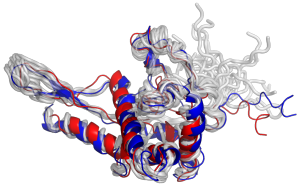
(a) 8DKE_P/8DKI_P



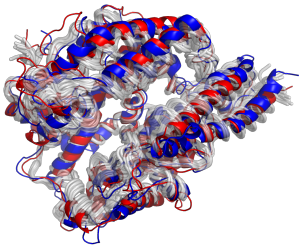
(b) 8G4C_D/8G4D_D



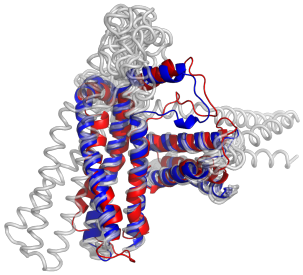
(c) 7ZF5_D/7ZF6_H



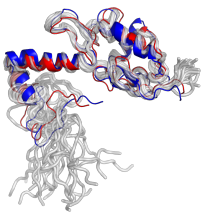
(d) 8HKX_AS7P/8HKY_AS7P



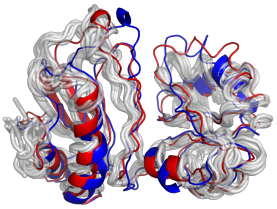
(e) 8HFX_E/8IFY_E



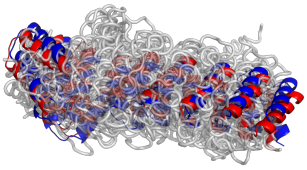
(f) 8SQZ_C/8SRM_C



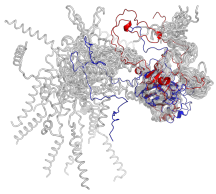
(g) 8HKX_S13P/8HKY_S13P



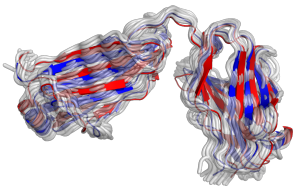
(h) 8HKY_AL1P/8HKZ_AL1P



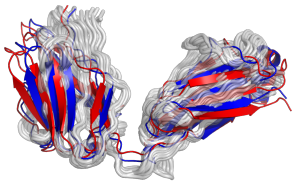
(i) 8V2D_a/8V3B_A



(j) 8E39_A/8E3A_A

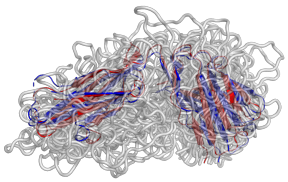


(k) 8TCA_L/8VEV_B

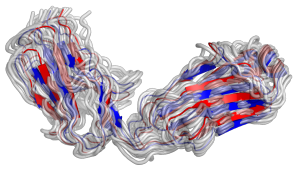


(l) 8EE5_L/8EF3_B

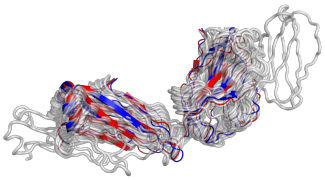
Fig. S13: Comparative visualization of PDB in **RAC-47** benchmark (Part 3)



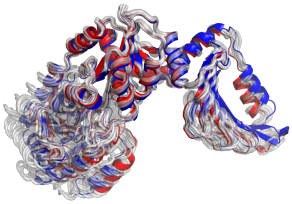
(a) 8EE5_H/8EF3_A



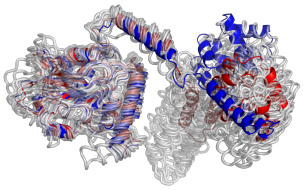
(b) 7ZF5_F/7ZF6_L



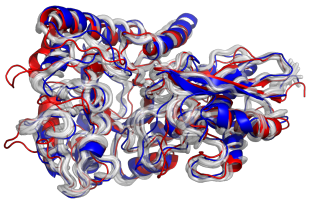
(c) 8HC3_E/8HC6_L



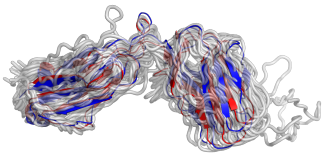
(d) 7QD7_AAA/7QH9_AAA



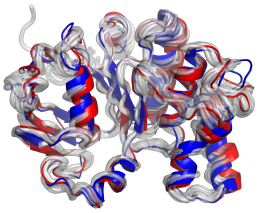
(e) 7T22_A/7T23_B



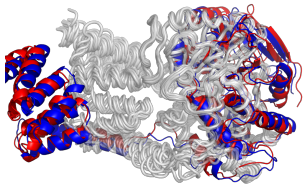
(f) 8UOP_A/8UOY_B



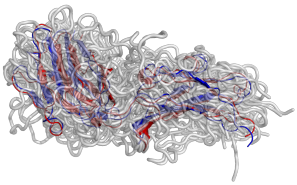
(g) 7S8G_L/7UDS_L



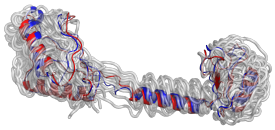
(h) 8I6O_B/8I6Q_B



(i) 8GAE_E/8GFT_E



(j) 8HC3_H/8HC5_H



(k) 8JSG_A/8JSH_A

Fig. S14: Comparative visualization of PDB in **RAC-47** benchmark (Part 4)