Controllable All-Atom Generation of Protein Sequence and Structure from Sequence-Only Inputs

Amy X. Lu[1,2*] **Wilson Yan**[1] **Sarah A. Robinson**[2] **Kevin K. Yang**[3] **Vladimir Gligorijevic**[2]

Kyunghyun Cho[2,4] **Richard Bonneau**[2] **Pieter Abbeel**[1] **Nathan Frey**[2]

[1]UC Berkeley [2]Prescient Design, Genentech [3]Microsoft Research [4]New York University

Abstract

We propose **PLAID (Protein Latent Induced Diffusion)**, a paradigm for generating all-atom structure and sequence of protein domains, by learning diffusions over the compressed latent space of pre-trained sequence-only input protein folding models. Since only sequence training data is required during generative model training, we augment the usable training dataset by $10^2\times$ to $10^4\times$ compared to other sequence-structure generative models. Further, this enlarges the annotations available for controllable generation, and we demonstrate compositional conditioning on function and organism, including a rich vocabulary of 2219 Gene Ontology functions. Samples exhibit cross-modal consistency, while possessing desired properties as measured by conditional Fréchet inception distance (FID). The PLAID paradigm avoids strong priors and massive imbalances from structure databases, scales readily with data and compute, and enables controllable generation of all-atom protein structures and sequences.

1 Introduction

Existing protein structure and sequence generation methods often treat sequence and structure as separate modalities, and most methods are either backbone-only, or require alternating between folding and inverse-folding steps for all-atom generation. For models capable of designing structure, the architectures and data representations (e.g., many equivariant architectures) are often not flexible and cannot leverage rapid progress in hardware-aware mechanisms for more scalable large language models. Evaluations often focus on characterizability and designability, with limited progress towards control of properties that can be experimentally tested in the wet lab. Structure generation methods that rely on experimentally-resolved structure databases are biased towards proteins that can be crystallized, whereas sequence-based databases have better representation across the viable protein distribution as traversed by evolution.

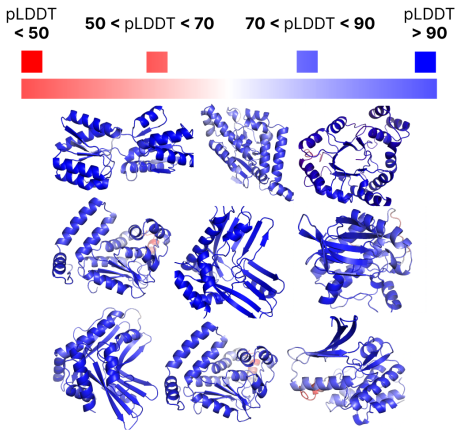

Figure 1: PLAID generates all-atom structure with high naturalness **despite not requiring structure during diffusion training.**

*Correspondence: amyxlu@berkeley.edu

Machine Learning for Structural Biology (MLSB) Workshop at the 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

	Generation Outputs			Conditioning		Training		
	Seq-uence	Side chain coords.	Backbone coords.	GO Term	Organism	Does not require structure	Co-generates without predictor	All protein families
MultiFlow	✓	✗	✓	✗	✗	✗	✓	✓
Protparadelle	✓	✓	✓	✗	✗	✗	✗	✓
ProteinGenerator	✓	✗	✓	✗	✗	✗	✗	✓
AbDiffuser	✓	✓	✓	✗	✗	✗	✓	✗
RFDiffusion	✗	✗	✓	✗	✗	✗	✗	✓
Genie	✗	✗	✓	✗	✗	✗	✗	✓
PLAID (ours)	✓	✓	✓	✓	✓	✓	✓	✓

Table 1: Comparison between model capabilities for selected co-generation models [6; 4; 7; 8] and backbone-only structure diffusion models [1; 9]. By defining the data distribution $p(\mathbf{x})$ as sequence, we gain more paired protein and annotation labels $\{\mathbf{x}, \mathbf{c}\}$ (e.g., GO terms).

Towards resolving these challenges, we introduce **Protein LAtent Induced Diffusion (PLAID)**, a latent diffusion model capable of **sequence and all-atom structure generation, while requiring only sequence data during training**. Since we can use sequence databases for all-atom generation, we augment the size of usable training data by a factor of $10^2\times$ to $10^4\times$, as compared to previous methods trained on non-synthetic data [1; 2; 3; 4], increases the availability of annotations for controllable generation, and allows us to focus on *protein domains*, portions of the protein with evolutionarily-conserved sequences that may have functional significance [5]. As a motivating demonstration, we examine two axis of compositional control: species of origin using 3617 organism taxons, and protein function using 2219 Gene Ontology terms.

2 Related Work

Denoising Diffusion Models and Latent Diffusion Diffusion models [10; 11; 12] approaches the generative modeling task of approximating the data distribution $p_\theta(\mathbf{x})$ by training a denoiser to remove iteratively added noise; during inference-time sampling, one starts from noise, and uses the denoiser to remove noise, until we arrive back at the data distribution, as defined by the training data. During training, models can be made conditional using classifier-free diffusion [10] by replacing the conditioning variable $\mathbf{c}$ with a null variable $\varnothing$ with some probability p_{uncond}. [2]

Multimodal Sequence-Structure and All-Atom Generation "All-atom generation" is the task of generating structure for amino acid side chains, as compared to "backbone only", which only includes carbon backbone atoms, and thus does not require the sequence to be defined. All-atom generation can be viewed as multimodal generation problem, where the 1D protein sequence and 3D protein structure are jointly produced.

Traditionally, computational protein design relies on Rosetta [13] to "pack" the side chains after backbone design. Early protein structure diffusion works followed the traditional approach by targeting the backbone-only setting [1; 2; 14; 9]. Some recent works have addressed this problem, but most require alternating between sequence and structure generation steps, and relies on a separately trained structure-prediction or sequence-prediction model. ProteinGenerator [7] uses diffusion on the one-hot encodings of sequences, and at each step, uses RosettaFold [15] to predict structure. Protpardelle [4] instead performs diffusion over structure, and at each step, uses ProteinMPNN [16] to predict sequence. Multiflow [6] takes a similar approach of sampling directly from the joint distribution rather than iterating between cross-modality prediction steps, but are unable to place the sidechain atoms. A tabular comparison can be found in Table 1. While the currently presented model does not perform conditioning by motif scaffolding and secondary structure, these information can

[2] The role that the training data plays in defining the data distribution $p_{\text{evolutionary}}(\mathbf{x})$ is an important motivator in this work. By restricting usable training data to structure datasets, the resulting model can only sample from $p_\theta(x_{\text{structure}})$, which only provides a small fraction of evolutionary diversity, is biased towards crystallizable proteins, and limits quantity of functional annotations that are available.

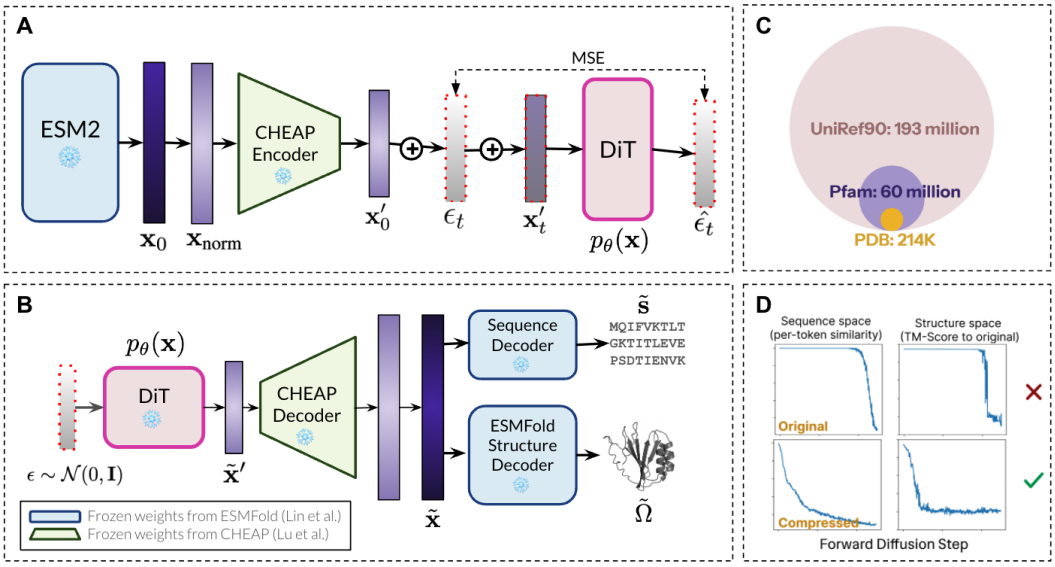

Figure 2: Overview of PLAID. **(A) Training.** PLAID performs diffusion in the ESMFold [3] latent space, to learn a representation of the joint distribution of protein sequence and structure using only sequence inputs. This embedding is compressed by frozen autoencoder weights [17]. **(B) Inference.** To obtain both sequence and structure at inference time, we first sample $\tilde{\mathbf{x}}' \in \mathbb{R}^{\frac{L}{2} \times 32}$, then decode back to $\tilde{x} \in \mathcal{X}$, which can be mapped back to sequence and structure with frozen decoders. **(C) Database sizes.** Drawn-to-scale comparison of the size of sequence databases (UniRef90 and Pfam) compared to structure-only datasets like the PDB. **(D) Latent space noise schedule.** Visualization of how a cosine noise schedule (orange, bottom) maps to corruptions (blue, top) in sequence (measured by per-token similarity, left) and structure (measured by backbone TM-score, right) spaces. Without the per-channel normalization and compression step, noise perturbations in the latent space do not map to a meaningful disruption in the sequence and structure space.

be gleaned from sequence data and conditioning capabilities can be readily extended in future work. Finally, to the best of our knowledge, no prior work has addressed all-atom generation of structure and sequence without requiring structural inputs to the generative model architecture.

3 PLAID: Protein Latent Induced Diffusion

3.1 Capturing the Joint Distribution of Sequence and Structure

Throughout this work, sequence is denoted as $\mathbf{s}$ while structure is denoted as Ω. The latent space of ESMFold [3] is denoted as $\mathbf{x} \in \mathbb{R}^{L \times 1024}$, with the valid space of protein sequence and structure abstractly denoted as $\mathbf{x} \in \mathcal{X}$. The compressed latent using the CHEAP [17] autoencoder $h(\mathbf{x})$ is denoted as $\mathbf{x}' \in \mathbb{R}^{\frac{L}{2} \times 32}$.

We begin with the motivation that sampling directly from $p(\mathbf{s}, \Omega)$ without implicitly factorizing it into $p(\Omega)p(\mathbf{s}|\Omega)$ (e.g. Protpardelle [4]) or $p(\Omega)p(\mathbf{s}|\Omega)$ (i.e. ProteinGenerator [7]) circumvents the difficulty in all-atom generation of knowing which side chain atoms to atoms place, since one can choose a latent manifold where the residues are not explicitly specified. Furthermore, avoiding iterative predictions across modalities and reliance on the prediction model is computationally cheaper, and closer to the underlying nature of protein data.

To characterize this joint distribution, we look to the latent space of protein folding models. This has the advantage of making use of the information captured in these models; as prediction models become increasingly multimodal and become capable of modeling structure complexes involving structure, nucleic acid, and small molecules [18; 19], being able to capture and diffuse in the latent space can easily be adapted to simultaneously generate more modalities. In this work, we choose

ESMFold because it is capable of predicting from sequence the all-atom structure, a capability that has seen more progress in *prediction* than *generation*.

Latent Space Compression Previous work [17] provides a way to characterize $p(\mathbf{s}, \Omega)$ by compressing the latent space of ESMFold [3], which predicts structure from single-sequence inputs. Figure 2A illustrates the pipeline during training, and Figure 2B describes how to obtain sequence and structure from latent. The encoder uses the same architecture and training procedure as in CHEAP [17], but is trained on Pfam to ensure in-distribution performance, and compresses inputs from $\mathbf{x} \in \mathbb{R}^{L \times 1024}$ to $\mathbf{x} \in \mathbb{R}^{\frac{L}{2} \times 32}$.

Reducing input size with an autoencoder alleviates computational costs for high resolution image modeling [20]. An issue with naively following a latent diffusion paradigm for protein generation is that the noise in the latent space may not map back the same way in the sequence and structure space. Figure 2C shows that without the normalization and compression post-processing steps in CHEAP, noise added in the latent space does not affect sequence and structure until the final timesteps in forward diffusion, despite using a cosine schedule (SNR and log-SNR curves shown below), meaning that the denoising task would be trivial for most sampled timesteps.

Architecture We use a Diffusion Transformer [21] (DiT) for the denoising task, which can be more flexibly finetuned for new input types than many equivariant networks in protein diffusion literature diffusion [22] and make use of rapidly growing research in hardware-aware speedups for Transformers during both training and inference [23; 24; 25], and found in early experiments that proportioning available memory to a larger DiT model was more helpful than using triangular self-attention [26]. We train our models using the xFormers [27] implementation of [28] which provided a 55.8% speedup with a 15.6% reduction in GPU memory usage in our inference-time benchmarking experiments compared to a vanilla implementation using PyTorch primitives.

3.2 Data and Training Details

Choice of Sequence Database The general paradigm in PLAID can be used on any sequence database, ranging from UniRef90 (193 million sequences) to metagenomic dataset such as BFD (2.5 billion sequences). We use Pfam because it provides more annotations for *in silico* evaluation, and because protein domains capture the range of functions well. More information can be found in Appendix A.

Function and Organism Conditioning Gene Ontology (GO) is a structured hierarchical vocabulary for annotating gene functions, biological processes, and cellular components across species [29; 30]. We examine all Pfam domains for which there exists a Gene Ontology mapping, of which there are 2219 total. All Pfam sequences are annotated with the full protein from which it was derived and the organism it came from in a 5 letter code. We examine all unique organisms in Pfam and find 3617 organisms. Models are trained with classifier-free guidance [31] with an unconditional dropout probability of $p_{\text{uncond}} = 0.3$). The model evaluated here has 2 billion parameters, trained for 800K steps.

Diffusion Training We use the discrete-time diffusion definition proposed in Ho et al. [10], using 1000 timesteps. Additional strategies are used to stabilize training and improve performance: min-SNR reweighting [32], v-diffusion [33; 34], self-conditioning [35; 36], and a Sigmoid noise schedule [37].

4 Results

Despite having never seen structures in its training data, PLAID generate structures with complex folds, such as TIM barrel patterns and beta strands, as seen in Figure 3C. Following prior work [1; 4; 6], we sample 100 proteins of lengths {100, 200, 300}. Cross-consistency (ccRMSD, ccTM) is unique to the setting where one samples directly from $p(\mathbf{s}, \Omega)$ (similar to Co-design 1) in Multiflow; however, since PLAID is an all-atom model, RMSD is measured for all atoms rather than $C\alpha$ atoms only. Self-consistency (scRMSD, scTM) comes from existing literature in backbone-only generation, and measures designability (which is less relevant in the case of multimodal generation,

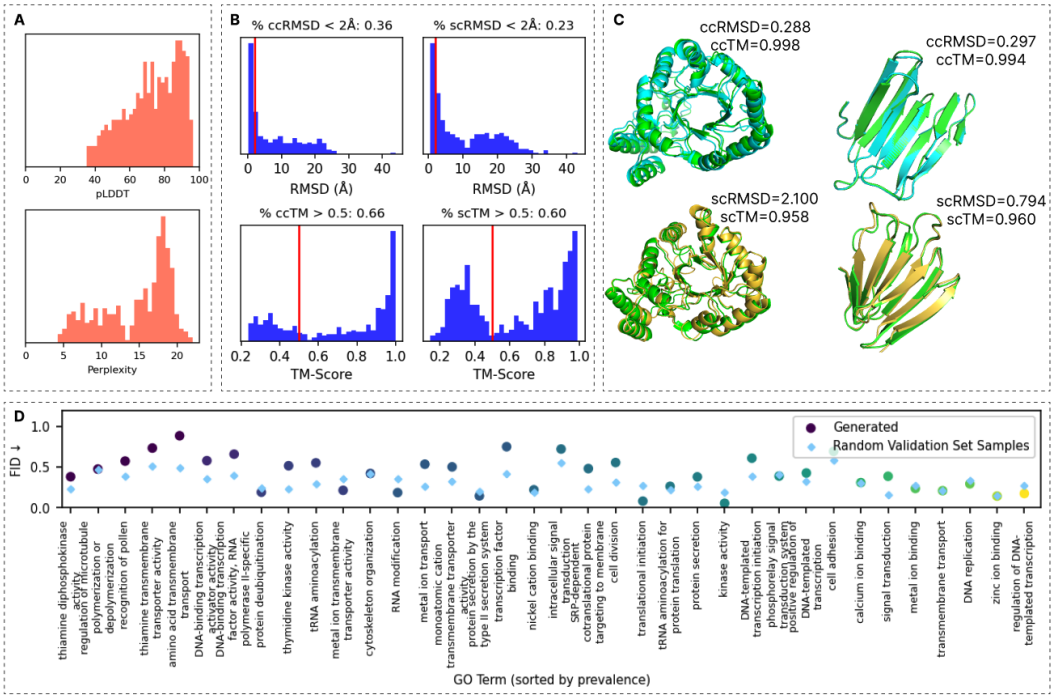

Figure 3: **PLAID generates high quality and complex folds without structural training data.** **(A, B)** Naturalness, cross consistency, and self-consistency. **(C)** Visual examples of samples, including TIM barrel-like (left) and beta sheets-like (right) folds. **(D)** Conditional generation by GO terms.

since ccRMSD serves a similar function). For structure construction for self-consistency, we use OmegaFold [38], another single-sequence structure predictor.

Figure 3A shows naturalness evaluations on samples. The pLDDT is returned naturally by the ESMFold decoder. The perplexity is evaluated under the autoregressive model, RITA XL [39]. Samples achieve good pLDDT and perplexity despite having not seen structures during training. Similarly, both cross-consistency and self-consistency metrics are strong, with 36% of generations achieving ccRMSD less than 2Å distance. Figure 3C provides a visual inspection. Figure 3D shows conditional generation results by GO terms; we choose a random subset of GO terms for which there are more than 512 samples in our holdout validation set, and compute the Frechet Inception Distance (FID) in the normalized and compressed latent space. For reference, the FID between a random tensor $\tilde{\mathbf{x}} \sim \mathcal{N}(0, 0.3)$ has an FID of 3.62 in this space.

5 Discussion

We proposed PLAID, a paradigm for multi-modal, controllable generation of proteins by diffusing in the latent space of a prediction model that maps single sequences to the desired modality; in this work, we examine all-atom generation of structure and sequence, and find that we can generate complex, high quality all-atom folds without using any structures during training. The performance is limited by the accuracy of the frozen decoder that is taken from the original prediction model. ESM3 [40] has shown that there is room for improvement in structure prediction from ESMFold [3], which both means that there are inefficiencies in the model that are propagated to PLAID, and that improved performance is scientifically achievable. Furthermore, since the structure decoder is deterministic, it is unable to sample different conformations in its current form. One way to achieve conformational diversity is to diffuse in the latent space of a generative model that returns a distribution over structural conformations instead. These limitations will be addressed in future work.

References

[1] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with RFdiffusion. *Nature*, 620:1089–1100, 2023.

[2] John Ingraham, Max Baranov, Zak Costello, Vincent Frappier, Ahmed Ismail, Shan Tie, Wujie Wang, Vincent Xue, Fritz Obermeyer, Andrew Beam, et al. Illuminating protein space with a programmable generative model. *bioRxiv*, 2022.12.01.518682, 2022.

[3] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[4] Alexander E Chu, Lucy Cheng, Gina El Nesr, Minkai Xu, and Po-Ssu Huang. An all-atom protein generative model. *bioRxiv*, 2023.

[5] Jaina Mistry, Sara Chuguransky, Lowri Williams, Matloob Qureshi, Gustavo A Salazar, Erik LL Sonnhammer, Silvio CE Tosatto, Lisanna Paladin, Shriya Raj, Lorna J Richardson, et al. Pfam: The protein families database in 2021. *Nucleic acids research*, 49(D1):D412–D419, 2021.

[6] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

[7] Sidney Lyayuga Lisanza, Jacob Merle Gershon, Sam Wayne Kenmore Tipps, Lucas Arnoldt, Samuel Hendel, Jeremiah Nelson Sims, Xinting Li, and David Baker. Joint generation of protein sequence and structure with RoseTTAFold sequence space diffusion. *bioRxiv*, 2023.

[8] Karolis Martinkus, Jan Ludwiczak, Wei-Ching Liang, Julien Lafrance-Vanasse, Isidro Hotzel, Arvind Rajpal, Yan Wu, Kyunghyun Cho, Richard Bonneau, Vladimir Gligorijevic, et al. Abdiffuser: full-atom generation of in-vitro functioning antibodies. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Yeqing Lin and Mohammed AlQuraishi. Generating novel, designable, and diverse protein structures by equivariantly diffusing oriented residue clouds. *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[11] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[12] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[13] Andrew Leaver-Fay, Michael Tyka, Steven M Lewis, Oliver F Lange, James Thompson, Ron Jacak, Kristian W Kaufman, P Douglas Renfrew, Colin A Smith, Will Sheffler, et al. Rosetta3: an object-oriented software suite for the simulation and design of macromolecules. In *Methods in enzymology*, volume 487, pages 545–574. Elsevier, 2011.

[14] Kevin E. Wu, Kevin K. Yang, Rianne van den Berg, James Y. Zou, Alex X. Lu, and Ava P. Amini. Protein structure generation via folding diffusion. *arXiv*, 2209.15611, 2022.

[15] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

[16] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 378:49–56, 2022.

[17] Amy X Lu, Wilson Yan, Kevin K Yang, Vladimir Gligorijevic, Kyunghyun Cho, Pieter Abbeel, Richard Bonneau, and Nathan Frey. Tokenized and continuous embedding compressions of protein sequence and structure. *bioRxiv*, pages 2024–08, 2024.

[18] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024.

[19] Rohith Krishna, Jue Wang, Woody Ahern, Pascal Sturmfels, Preetham Venkatesh, Indrek Kalvet, Gyu Rie Lee, Felix S Morey-Burrows, Ivan Anishchenko, Ian R Humphreys, et al. Generalized biomolecular modeling and design with rosettafold all-atom. *Science*, 384(6693):eadl2528, 2024.

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[21] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[22] Jason Yim, Brian L Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. SE (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023.

[23] Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2022.

[24] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

[25] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[26] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

[27] Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library. `https://github.com/facebookresearch/xformers`, 2022.

[28] Markus N Rabe and Charles Staats. Self-attention does not need $o(n^2)$ memory. *arXiv preprint arXiv:2112.05682*, 2021.

[29] TGO Consortium, Suzi A Aleksander, James Balhoff, Seth Carbon, JM Cherry, Harold J Drabkin, Dustin Ebert, Marc Feuermann, Pascale Gaudet, and Nomi L Harris. The gene ontology knowledgebase in 2023. *Genetics*, 224(1):iyad031, 2023.

[30] Michael Ashburner, Catherine A Ball, Judith A Blake, David Botstein, Heather Butler, J Michael Cherry, Allan P Davis, Kara Dolinski, Selina S Dwight, Janan T Eppig, et al. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

[31] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[32] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7441–7451, 2023.

[33] Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 5404–5411, 2024.

[34] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[35] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[36] Allan Jabri, David Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. *arXiv preprint arXiv:2212.11972*, 2022.

[37] Ting Chen. On the importance of noise scheduling for diffusion models. *arXiv preprint arXiv:2301.10972*, 2023.

[38] Ruidong Wu, Fan Ding, Rui Wang, Rui Shen, Xiwen Zhang, Shitong Luo, Chenpeng Su, Zuofan Wu, Qi Xie, Bonnie Berger, et al. High-resolution de novo structure prediction from primary sequence. *bioRxiv*, 2022.

[39] Daniel Hesslow, Niccoló Zanichelli, Pascal Notin, Iacopo Poli, and Debora Marks. RITA: a study on scaling up generative protein sequence models. *arXiv preprint arXiv:2205.05789*, 2022.

[40] Tomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *bioRxiv*, pages 2024–07, 2024.

[41] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[42] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[43] Ian Sillitoe, Nicola Bordin, Natalie Dawson, Vaishali P Waman, Paul Ashford, Harry M Scholes, Camilla SM Pang, Laurel Woodridge, Clemens Rauer, Neeladri Sen, et al. Cath: increased structural coverage of functional space. *Nucleic acids research*, 49(D1):D266–D273, 2021.

Appendix

A Data

Approximately 46.7% of the dataset (N=24,637,236) is annotated with a GO term. Using the publicly available mapping as of July 1, 2024, we take a count of all GO occurences; for each Pfam entry with multiple GO entries, we pick the one with the fewest GO occurences to encourage more descriptive and distinct GO labels.

B Sampling

Inference-time sampling hyperparameters provides the user with additional control over quality and sampling speed trade-off. PLAID supports the DDPM sampler [10] and the DDIM sampler [41], as well as the improved speed samplers from DPM++ [42]. We find that using the DDIM sampler with 500 timesteps using either the sigmoid or cosine schedulers works best during inference, and reasonable samples can be obtained using the DPM++2M-SDE sampler with only 20 steps. Experiments shown here uses DDIM sampler with the sigmoid noise schedule at 500 timesteps.

Note that the performance bottleneck is found mostly during the latent sampling and structure decoding (which depends on the number of recycling iterations [26; 3] used); however, these two processes can be easily decoupled and parallelized, which cannot be done in existing protein diffusion methods. Furthermore, it allows us to prefilter which latents to decode using heuristic methods, and decode only those latents to structure, which would boost performance for nearly the same computational cost. We do not empirically explore this in this paper to provide a fair comparison, and because the filtering criteria would vary greatly by downstream use.

C Additional Figures

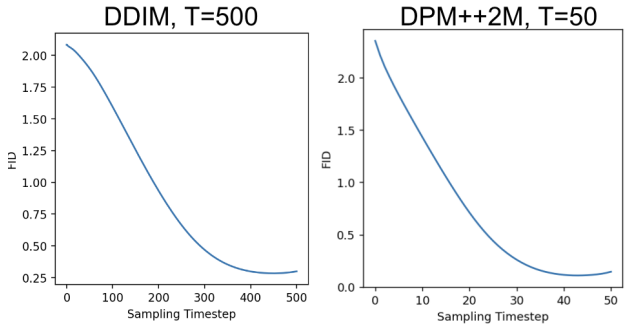

Figure 4: FID across sampling (reverse diffusion) timesteps for the DDIM [41] sampler and the DPM++2M [42] sampler. For both, sample quality decreases steadily over time before plateauing. DPM++2M can achieve low FID results with only 10% of the original number of steps, but final results are still slightly worse.

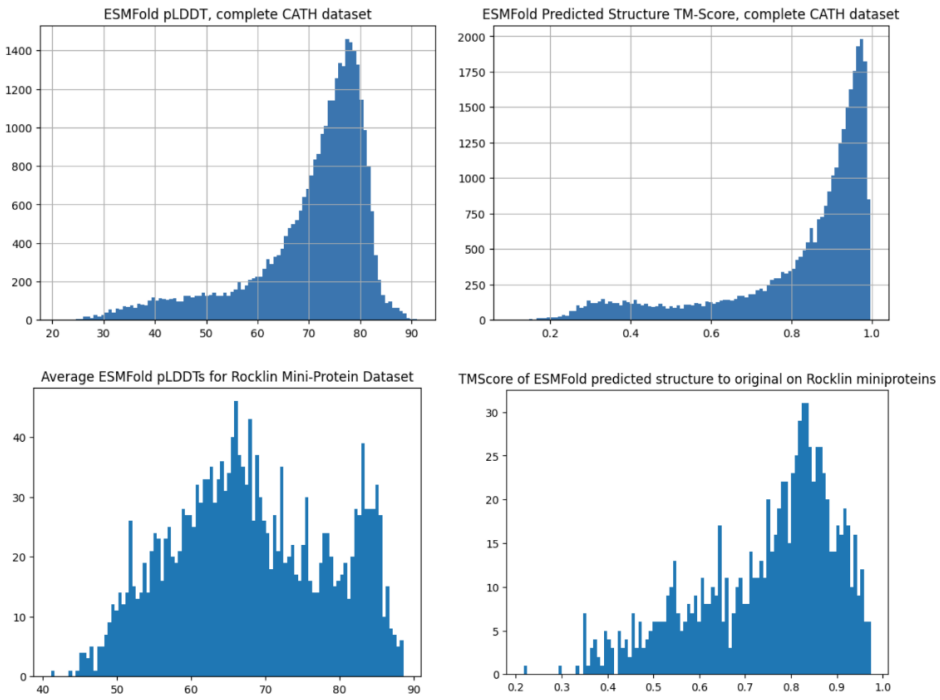

Figure 5: ESMFold pLDDT and TM-Scores on the complete Rocklin mini-proteins dataset and the CATH dataset, as a gauge for how ESMFold performs on different distributions, and what sorts of settings are most in-distribution for the ESMFold structure decoder.

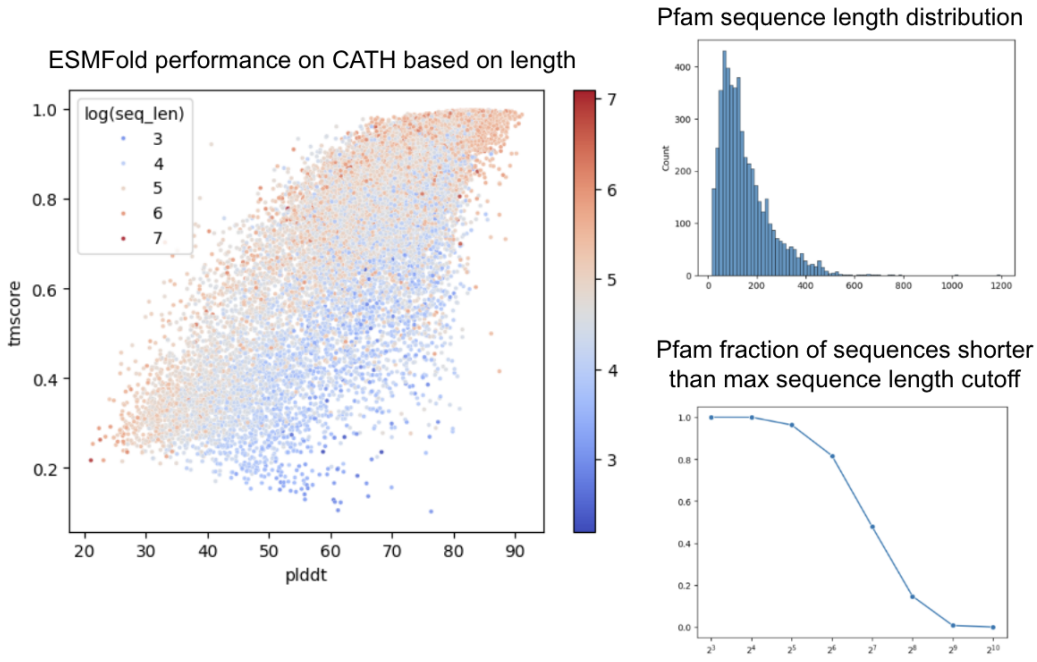

Figure 7: **Left:** Understanding length generalization for ESMFold to understand how and why sampler quality changes at different timesteps, and whether or not pLDDT is a trustworthy evaluation metric, by running ESMFold over the CATH [43] dataset. Length is an important determinant of model "hallucination"; for shorter sequences, ESMFold can be quite confident yet wrong. **(Right:)** Understanding which length cutoff to use during training. We use a cut off of 512 based on the findings here, since 512 would allow us to see full domains for most sequences in the dataset, while still having manageable GPU memory needs.