forge: sequence-based binder design with latent flow matching

Young Su Ko **Wei Wang**

Department of Chemistry and Biochemistry
University of California, San Diego
{y4ko, wei-wang}@ucsd.edu

Abstract

We present `forge-v0`, a sequence-based flow matching model for protein binder design. Drawing parallels from text-to-image generation, we treat binder design as a target-to-binder generation task. Rather than the text encoder and variational autoencoder typically used for text and images respectively, we leverage Raygun, a pre-trained autoencoder for protein sequence to represent both the target and binder. Using a state-of-the-art flow matching architecture, we trained `forge-v0` on $\sim$10M protein-protein interactions from the STRING database, rather than complex structures from the Protein Data Bank, learning a broader distribution of natural interactions. In our proof-of-concept evaluation, we performed an in silico binder design benchmark, in which `forge-v0` generated binders with higher structure-based metrics (ipTM and ipSAE) than RFDiffusion on 10/11 targets. These results motivate further development of `forge` towards designing and validating binders against therapeutically relevant targets out of reach for structure-based approaches.

1 Introduction

Binder design is central to numerous protein engineering tasks [1]. With advances in deep learning (DL), structure-based binder design methods such as RFDiffusion/ProteinMPNN [2] and BindCraft [3] have demonstrated the ability to de novo design binders. Underlying their success is the Protein Data Bank (PDB), a database of protein structures from which models learn how proteins interact in 3D. While structure provides a strong inductive bias, it can also be a limiting factor as not all proteins form stable tertiary conformations.

As emphasized by Alamdari et al. [4] in EvoDiff, proteins deposited in the PDB represent a small subset of the natural protein space. They are heavily biased toward proteins of high interest to structural biologists, given the low-throughput nature of structural determination, and those that are stable and crystallizable [5] (Figure 1A). Inheriting the biases present in the training data, structure-based methods capably design binders for PDB-like proteins but struggle for those out-of-distribution. Critically, many therapeutically relevant targets, such as transcription factors and fusion oncoproteins, are proteins that lie outside the PDB-like protein distribution [6].

The sequence-structure-function paradigm states that the sequence of a protein determines its structure, which determines its function. Given the aforementioned limitations of structure, a promising avenue is to exploit the indirect sequence-to-function relationship. Indeed, sequence-based methods like PepMLM [7] have successfully designed functional proteins (peptide binders) from sequence alone. Here, we look to extend the success of sequence-based peptide design to general protein binder design, which remains relatively underexplored.

Machine Learning for Structural Biology Workshop

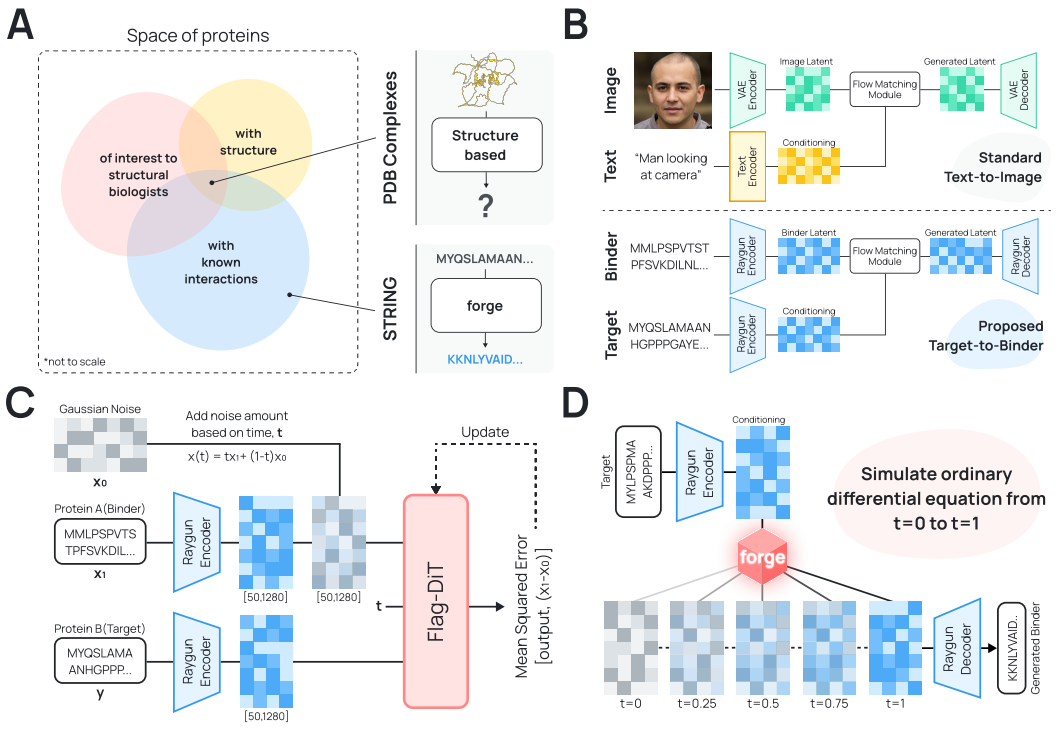

Figure 1: (**A**) The foundation for deep-learning based binder design, the training data, differs between structure-based and sequence-based methods. The PDB complex structures used to train structure-based methods represent the intersection of proteins of interest to structural biologists, that are crystallizable, and have a solved complex. As a result, structure-based methods can struggle for targets that are out-of-distribution, such as heavily disordered proteins. On the other hand, sequence-based methods can leverage protein-interaction databases such as STRING, which represent the known interactions but are not limited by structure. (**B**) By replacing the variational autoencoder and text encoder of text-to-image (T2I) architectures with Raygun, adapting T2I models for target-to-binder generation is straightforward. (**C**) **Training.** Interacting protein pairs are embedded into Raygun latent space. Noise is added to the binder embedding at a sampled timepoint $t \sim \mathcal{U}[0, 1)$, and the Flag-DiT backbone predicts the velocity toward the clean binder embedding, conditioned on the target embedding. (**D**) **Inference**. We encode the target sequence with Raygun and draw noise $X_0 \sim \mathcal{N}(0, I_d)$ in latent space. With the trained `forge` model, we simulate the learned ordinary differential equation to transform noise into a binder latent, which is decoded by Raygun back to sequence.

In addition to bypassing structure, a key motivator for a sequence-based binder design is the ability to learn from a wider subset of natural interactions. While structure-based methods learn from protein complex structures, sequence-based methods can leverage much larger protein-protein interaction (PPI) databases like STRING [8], which contain nearly 100M interactions, even after sequence-based clustering [9]. Importantly, many of these PPIs are detected in a high-throughput manner with methods like co-immunoprecipitation and yeast two-hybrid [10] that do not require structural determination. As a result, sequence-based PPI databases not only have more data but also interactions between proteins unable to be captured structurally. By learning directly from these diverse natural interactions, a sequence-based model may be able to more effectively generate binders for a wider range of targets.

To train such a model, rather than starting from scratch, we draw inspiration from the well-established text-to-image (T2I) generation task. In T2I, flow matching models such as Lumina-T2I [11] are state-of-the-art (SoTA), using classifier-free guidance (CFG) [12] to condition on text-encoder embeddings of captions to generate corresponding images. Thus, a cost-effective and practical strategy is to adapt Lumina-T2I for target-to-binder generation. To do so, we need the protein equivalents of

the variational autoencoder (VAE) used to represent images and the text encoder used to represent conditioning text. Raygun, a recent method for protein design, can conveniently serve as both.

Raygun is a pre-trained autoencoder that maps proteins of arbitrary length into a fixed 50×1280 latent representation [13]. Similar to how VAEs provide a smooth latent space that captures the high-level semantic information of images, Raygun provides a similar purpose, for example, producing latents that better capture the CATH hierarchies than standard ESM2 latents [13]. Because target-to-binder generation uses protein sequences as the conditioning input, Raygun naturally serves as the analogue for both the VAE and text-encoder of T2I. With Raygun, adapting Lumina-T2I for protein binder design becomes straightforward (Figure 1B).

Distilling the principles outlined above, we present our prototype `forge-v0` (flow-matching on Raygun embeddings), a sequence-based latent flow matching model trained on ∼10M PPIs for smithing protein binders. By learning from a broader and more diverse set of natural interactions, `forge-v0` seeks to fill the gap for designing binders for targets that lie outside the PDB-like regime. Initial in silico results are encouraging, motivating further refinement, benchmarking, and ultimately, experimental validation.

2 Methods

2.1 Flow Matching

Flow matching [14] (FM) learns a time-dependent vector field, u_t^θ, that transports samples from a simple prior distribution (e.g., Gaussian noise) to the data distribution. We use conditional flow matching (CFM), which provides a tractable training objective [15]. Specifically, given a data sample $X_1 \sim q$ and a noise sample $X_0 \sim \mathcal{N}(0, I_d)$, we define the path from noise to data as the linear interpolation $X_t = (1-t)X_0 + tX_1$. The model u_t^θ is trained by predicting the target velocity along this path, which simplifies to $X_1 - X_0$: $\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,X_0,X_1} \|u_t^\theta(X_t, t) - (X_1 - X_0)\|^2$. After training, new samples are generated by drawing $X_0 \sim \mathcal{N}(0, I_d)$ and simulating the ordinary differential equation (ODE) $\frac{dX_t}{dt} = u_t^\theta(X_t, t)$ from $t = 0$ to $t = 1$. See Appendix A.1 for additional details.

2.1.1 Classifier-Free Guidance

To enable conditional generation–generating a binder conditioned on a target protein–we use classifier-free guidance (CFG) [12]. A single network $u_t^\theta(x \mid y)$ is trained on paired samples (X_1, y), where y is the corresponding conditioning information, here, the interacting protein partner. During training, y is dropped to a null token $\varnothing$ with probability η such that the model is exposed to both conditional and unconditional modes. At inference, the vector fields are interpolated as $\tilde{u}_t(x \mid y) = u_t^\theta(x \mid \varnothing) + \omega\big(u_t^\theta(x \mid y) - u_t^\theta(x \mid \varnothing)\big)$, where ω is the guidance scale. By increasing ω, greater weight is put onto removing the signal from the unconditional model. However, much like how T2I models use text encoder embeddings rather than a discrete class token to generalize to unseen prompts, `forge-v0` uses Raygun embeddings to represent conditioning proteins [16, 11]. See Appendix A.2 for additional details.

2.2 `forge-v0`

`forge-v0` is a 531M parameter flow-matching model for binder design, operating in the latent space of Raygun. It is trained on ∼5M interaction pairs (or ∼10M interactions) from ∼3M unique sequences. `forge-v0` adapts the Flow-based Large Diffusion Transformer (Flag-DiT) backbone introduced by Lumina-T2X [11]. An overview of the training and sampling procedures is shown in Figure 1C,D. Additional details regarding dataset pre-processing and model architecture are moved to Appendix B.3.

2.3 Benchmarking

2.3.1 In silico Evaluation

Generative models of proteins can be evaluated on a number of dimensions [17]. However, for binder design, the most important criterion is the ability to generate binders. While future work will

	TrkA	LTK	IL2Ra	IL10Ra	SARS-CoV-2 RBD	PD-L1	VirB8	FGFR2	MDM2	InsulinR	IL-7Ra
BindCraft	0.855 ± 0.111	0.837 ± 0.142	0.877 ± 0.095	0.644 ± 0.223	0.643 ± 0.211	0.923 ± 0.049	0.847 ± 0.141	0.840 ± 0.106	0.916 ± 0.067	0.890 ± 0.107	0.657 ± 0.195
DSM	0.831 ± 0.158	0.853 ± 0.145	0.820 ± 0.153	0.596 ± 0.233	0.644 ± 0.196	0.904 ± 0.094	0.888 ± 0.093	0.839 ± 0.104	0.890 ± 0.077	0.784 ± 0.144	0.524 ± 0.196
RFDiffusion	0.832 ± 0.132	0.842 ± 0.149	0.806 ± 0.139	0.509 ± 0.180	0.521 ± 0.234	0.891 ± 0.064	0.849 ± 0.109	0.791 ± 0.125	0.896 ± 0.081	0.679 ± 0.210	0.581 ± 0.185
`forge-v0`	0.866 ± 0.097	0.865 ± 0.140	0.859 ± 0.113	0.691 ± 0.179	0.626 ± 0.200	0.919 ± 0.051	0.862 ± 0.111	0.831 ± 0.108	0.861 ± 0.114	0.829 ± 0.112	0.632 ± 0.181

	TrkA	LTK	IL2Ra	IL10Ra	SARS-CoV-2 RBD	PD-L1	VirB8	FGFR2	MDM2	InsulinR	IL-7Ra
BindCraft	0.525 ± 0.252	0.393 ± 0.287	0.542 ± 0.235	0.179 ± 0.216	0.190 ± 0.251	0.727 ± 0.155	0.471 ± 0.281	0.524 ± 0.239	0.700 ± 0.225	0.630 ± 0.249	0.182 ± 0.211
DSM	0.540 ± 0.271	0.443 ± 0.248	0.443 ± 0.246	0.146 ± 0.169	0.157 ± 0.189	0.683 ± 0.209	0.573 ± 0.231	0.502 ± 0.216	0.571 ± 0.231	0.351 ± 0.216	0.078 ± 0.118
RFDiffusion	0.477 ± 0.263	0.402 ± 0.258	0.369 ± 0.243	0.045 ± 0.069	0.100 ± 0.187	0.584 ± 0.231	0.436 ± 0.258	0.394 ± 0.241	0.661 ± 0.229	0.242 ± 0.261	0.106 ± 0.137
`forge-v0`	0.569 ± 0.227	0.474 ± 0.245	0.510 ± 0.231	0.209 ± 0.191	0.132 ± 0.158	0.697 ± 0.170	0.498 ± 0.252	0.471 ± 0.240	0.478 ± 0.256	0.410 ± 0.218	0.142 ± 0.142

Table 1: **Mean ± standard deviation of ipTM (top) and ipSAE (bottom) per target and model.** Best mean per target/metric is highlighted in and second-best in .

assess other metrics to add nuance, for this prototype, we focus on investigating how well `forge-v0` generates binders compared to existing methods.

However, it is non-trivial to meaningfully benchmark generative methods for binder design. First, although the ideal benchmark would involve generating N binders per method and experimentally measuring the binding affinity to multiple targets, the prohibitive cost of large-scale experimental validation makes it infeasible. While in silico proxies for binding are a natural solution, the second hurdle stems from the fact that no in silico metric has been shown to definitively predict binding [18]. Despite these limitations, recent works have highlighted two metrics computed from predicted complex structures: the interface predicted template modeling (ipTM) [19] and interface prediction from aligned errors (ipSAE) [20] scores. While ipSAE (specifically ipSAE_min, see Appendix D) is stated to be more discriminative than ipTM [20], ipTM has been successfully used in BindCraft [3]–we report both. For predicting the complex structure, while AlphaFold3 [21] is state-of-the-art, we use Boltz-2 [22] as a competitive open-source alternative. While we make the assumption that higher ipTM and ipSAE scores correspond to "better" binders, we also discuss the shortcomings in the Conclusion.

2.3.2 Selected Models and Targets

We compare four methods: two structure-based (RFDiffusion with ProteinMPNN [2], BindCraft (specifically the AlphaFold-trajectory generation step) [3]) and two sequence-based (Diffusion Sequence Model (DSM) [23], `forge-v0`). We explain their justifications, as well as a brief overview and sampling procedure for each model in Appendix C.

From the meta-analysis of de novo binders by Overath et al. [24], we selected 11 of the 15 diverse targets for which prior methods have successfully designed binders. While future evaluation will focus on targets that pressure-test structure-based methods, these 11 common targets serve as practical starting point for evaluating both structure and sequence-based methods on equal footing. Additional details including sequences and PDB IDs are provided in Table 3.

3 Results

For each method, we generate 100 50AA-long binder sequences per target and predict their complexes with Boltz-2. While scoring and filtering down thousands of generated binders is typical, we are specifically interested in evaluating the generative capabilities of each model. In other words, to measure how well a model inherently generates binders, as opposed to how well a filter may discover a binder by chance, we do not perform any removal or selection before complex prediction to account for this confounding variable.

3.1 Sequence-based Binder Design Rivals Structure-based

We first compare `forge-v0` with the structure-based baselines (Table 1). Overall, BindCraft achieves the highest average ipTM for 6 targets and the highest average ipSAE for 7 targets. The strong performance of BindCraft is expected as it directly optimizes for structural metrics during generation. However, compared to RFDiffusion, a fairer baseline, `forge-v0` achieves a higher average ipTM and ipSAE for 10 out of 11 targets. Compared to the sequence-based DSM, `forge-v0` achieves higher mean scores on both metrics for 7 targets. In summary, `forge-v0` produced the best mean in silico scores for 3 targets (TrkA, LTK, IL10Ra) and second best for 5 others.

In this filter-free setting, sequence-based methods natively generate binders with higher ipTM and ipSAE than RFDiffusion. Second, we see that even though both DSM and `forge-v0` are sequence-based, the differences in training protocol lead to different results. While in silico metrics have their limitations, the results encourage a continued development of `forge` towards the more definitive experimental benchmark.

3.2 Investigating the Failure Modes of `forge-v0`

In binder design, some targets are considered "harder" than others. For example, a lack of stable structure can make a target challenging for structure-based methods. However, for sequence-based methods, it is not immediately clear why we have higher average ipTM/ipSAE for some targets over others. A simple explanation would be that performance correlates with the number of similar datapoints in the training data. For each target, we ran `mmseqs easy-search` with default parameters to identify similar sequences in the training set and counted the total number of PPIs involving those similar sequences for both the `forge-v0` and DSM training sets.

However, we did not observe a simple relationship between training coverage and performance (Table 2). Both DSM and `forge-v0` generate low-scoring binders for SARS-CoV-2 RBD, which is in line with expectations, as the RBD has no training set homologs. However, DSM achieves the strongest scores on VirB8 despite zero homologous sequence interactions. `forge-v0` outperforms DSM on FGFR2 even though DSM sees more than three times as many FGFR2-related interactions.

Another possibility is that sequence-based methods generate un-PDB-like binder for certain targets. For a given target, if STRING primarily captures interactions with proteins absent from the PDB, this can lead to generating proteins out of distribution for the structure predictors, resulting in a lower ipTM or ipSAE. We will further examine this possibility, calculating the maximum sequence identity of training data to PDB structures.

	FGFR2	IL2Ra	IL7Ra	IL10Ra	InsulinR	LTK	MDM2	PD-L1	SARS-CoV-2 RBD	TrkA	VirB8
DSM training set											
Homologs	58	1	4	3	14	5	4	6	0	10	0
Interactions	782	28	72	34	504	32	100	50	0	192	0
forge-v0 training set											
Homologs	116	48	128	39	115	20	28	71	0	18	3
Interactions	248	115	237	63	242	26	32	99	0	29	3

Table 2: **Overlap of benchmark targets with training data.** For each benchmark target, we report the number of homologous proteins and the total number of training interactions involving any homolog.

4 Conclusion

We introduced `forge-v0`, a sequence-based latent flow matching model for binder design. The results, in addition to encouraging further refinement of `forge`, highlight a hurdle for sequence-based binder design: the existing infrastructure for evaluation is predominantly structure-based. Sequence-based methods, by learning from wider range of interactions, many of which are absent from the PDB, can generate sequences that are outside distribution of structures used to train the structure predictors. In these cases, lower structure-based metrics may simply capture the inability to predict the complex, rather than meaningful information about binder quality. **In other words, structure-based metrics may not be the appropriate ruler for measuring the success of a sequence-based model.**

Thus, a major future direction is to build out the appropriate infrastructure for evaluating sequence-based binder designs. Sequence-based PPI prediction is a promising direction, but will need to demonstrate the ability to predict the interactions between de novo designed proteins. Rather than aiming for the highest accuracy, approaches like MINT [9], which aim to learn the "language of interactions" provide a promising foundation.

Looking forward, we outline the priorities for `forge-v1`. First, refining model training. While the current prototype was evaluated on validation loss, a lower validation loss does not guarantee higher quality samples. As `forge` relies on Raygun to decode latents, it is beneficial for the generated latents

to remain in-distribution with respect to the decoder. A practical approach is to compute the Fréchet distance between generated latents [25, 4] and the validation-set Raygun embeddings during each validation step.

Second, expanding in silico evaluation. This includes investigating the effects of sampling hyperparameters (e.g. integration steps, guidance scales), incorporating recently developed methods [26, 27, 28], evaluating performance across binder lengths, and examining end-to-end pipelines rather than just the generative step.

Lastly, experimental validation: designing a binder for a target that lies far outside the PDB-like regime can illustrate the unique advantage of sequence-based binder design.

References

[1] Daniel R. Fox, Cyntia Taveneau, Janik Clement, Rhys Grinter, and Gavin J. Knott. Code to complex: AI-driven de novo binder design. *Structure*, page S0969212625003119, September 2025.

[2] 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, Basile I. M. Wicky, Nikita Hanikel, Samuel J. Pellock, Alexis Courbet, William Sheffler, Jue Wang, Preetham Venkatesh, Isaac Sappington, Susana Vázquez Torres, Anna Lauko, Valentin De Bortoli, Emile Mathieu, Sergey Ovchinnikov, Regina Barzilay, Tommi S. Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. De novo design of protein structure and function with RFdiffusion. *Nature*, 620(7976):1089–1100, August 2023.

[3] Martin Pacesa, Lennart Nickel, Christian Schellhaas, Joseph Schmidt, Ekaterina Pyatova, Lucas Kissling, Patrick Barendse, Jagrity Choudhury, Srajan Kapoor, Ana Alcaraz-Serna, Yehlin Cho, Kourosh H. Ghamary, Laura Vinué, Brahm J. Yachnin, Andrew M. Wollacott, Stephen Buckley, Adrie H. Westphal, Simon Lindhoud, Sandrine Georgeon, Casper A. Goverde, Georgios N. Hatzopoulos, Pierre Gönczy, Yannick D. Muller, Gerald Schwank, Daan C. Swarts, Alex J. Vecchio, Bernard L. Schneider, Sergey Ovchinnikov, and Bruno E. Correia. One-shot design of functional protein binders with BindCraft. *Nature*, August 2025.

[4] Sarah Alamdari, Nitya Thakkar, Rianne Van Den Berg, Neil Tenenholtz, Robert Strome, Alan M. Moses, Alex X. Lu, Nicolò Fusi, Ava P. Amini, and Kevin K. Yang. Protein generation with evolutionary diffusion: sequence is all you need, September 2023.

[5] Johannes Kirchmair, Patrick Markt, Simona Distinto, Daniela Schuster, Gudrun M. Spitzer, Klaus R. Liedl, Thierry Langer, and Gerhard Wolber. The Protein Data Bank (PDB), Its Related Services and Software Tools as Key Components for In Silico Guided Drug Discovery. *Journal of Medicinal Chemistry*, 51(22):7021–7040, November 2008.

[6] Tianlai Chen, Lauren Hong, Vivian Yudistyra, Sophia Vincoff, and Pranam Chatterjee. Generative design of therapeutics that bind and modulate protein states. *Current Opinion in Biomedical Engineering*, 28:100496, December 2023.

[7] Leo Tianlai Chen, Zachary Quinn, Madeleine Dumas, Christina Peng, Lauren Hong, Moises Lopez-Gonzalez, Alexander Mestre, Rio Watson, Sophia Vincoff, Lin Zhao, Jianli Wu, Audrey Stavrand, Mayumi Schaepers-Cheu, Tian Zi Wang, Divya Srijay, Connor Monticello, Pranay Vure, Rishab Pulugurta, Sarah Pertsemlidis, Kseniia Kholina, Shrey Goel, Matthew P. DeLisa, Jen-Tsan Ashley Chi, Ray Truant, Hector C. Aguilar, and Pranam Chatterjee. Target sequence-conditioned design of peptide binders using masked language modeling. *Nature Biotechnology*, August 2025.

[8] Damian Szklarczyk, Katerina Nastou, Mikaela Koutrouli, Rebecca Kirsch, Farrokh Mehryary, Radja Hachilif, Dewei Hu, Matteo E Peluso, Qingyao Huang, Tao Fang, Nadezhda T Doncheva, Sampo Pyysalo, Peer Bork, Lars J Jensen, and Christian von Mering. The STRING database in 2025: protein networks with directionality of regulation. *Nucleic Acids Research*, 53(D1):D730–D737, January 2025.

[9] Varun Ullanat, Bowen Jing, Samuel Sledzieski, and Bonnie Berger. Learning the language of protein-protein interactions, March 2025.

[10] Michael Tanowitz and Mark Von Zastrow. Identification of Protein Interactions by Yeast Two-Hybrid Screening and Coimmunoprecipitation. In *Receptor Signal Transduction Protocols*, volume 259, pages 353–370. Humana Press, New Jersey, March 2004.

[11] Peng Gao, Le Zhuo, Dongyang Liu, Ruoyi Du, Xu Luo, Longtian Qiu, Yuhang Zhang, Chen Lin, Rongjie Huang, Shijie Geng, Renrui Zhang, Junlin Xi, Wenqi Shao, Zhengkai Jiang, Tianshuo Yang, Weicai Ye, He Tong, Jingwen He, Yu Qiao, and Hongsheng Li. Lumina-T2X: Transforming Text into Any Modality, Resolution, and Duration via Flow-based Large Diffusion Transformers, June 2024. arXiv:2405.05945 [cs].

[12] Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance, July 2022. arXiv:2207.12598 [cs].

[13] Kapil Devkota, Daichi Shonai, Joey Mao, Young Su Ko, Wei Wang, Scott Soderling, and Rohit Singh. Miniaturizing, Modifying, and Magnifying Nature's Proteins with Raygun, August 2024.

[14] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow Matching Guide and Code, December 2024. arXiv:2412.06264 [cs].

[15] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling, February 2023. arXiv:2210.02747 [cs].

[16] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding, May 2022. arXiv:2205.11487 [cs].

[17] Pavel Strashnov, Andrey Shevtsov, Viacheslav Meshchaninov, Maria Ivanova, Fedor Nikolaev, Olga Kardymon, and Dmitry Vetrov. Towards Robust Evaluation of Protein Generative Models: A Systematic Analysis of Metrics, October 2024.

[18] Tudor-Stefan Cotet, Igor Krawczuk, Filippo Stocco, Noelia Ferruz, Anthony Gitter, Yoichi Kurumida, Lucas De Almeida Machado, Francesco Paesani, Cianna N. Calia, Chance A. Challacombe, Nikhil Haas, Ahmad Qamar, Bruno E. Correia, Martin Pacesa, Lennart Nickel, Kartic Subr, Leonardo V. Castorina, Maxwell J. Campbell, Constance Ferragu, Patrick Kidger, Logan Hallee, Christopher W. Wood, Michael J. Stam, Tadas Kluonis, Süleyman Mert Ünal, Elian Belot, Alexander Naka, and Adaptyv Competition Organizers. Crowdsourced Protein Design: Lessons From the Adaptyv EGFR Binder Competition, April 2025.

[19] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021.

[20] Roland L. Dunbrack. *Rēs ipSAE loquunt* : What's wrong with AlphaFold's *ipTM* score and how to fix it, February 2025.

[21] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, June 2024.

[22] Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, David Kwabi-Addo, Dominique Beaini, Tommi Jaakkola, and Regina Barzilay. Boltz-2: Towards Accurate and Efficient Binding Affinity Prediction, June 2025.

[23] Logan Hallee, Nikolaos Rafailidis, David B. Bichara, and Jason P. Gleghorn. Diffusion Sequence Models for Enhanced Protein Representation and Generation, June 2025. arXiv:2506.08293 [q-bio].

[24] Max D. Overath, Andreas S. H. Rygaard, Christian P. Jacobsen, Valentas Brasas, Oliver Morell, Pietro Sormanni, and Timothy P. Jenkins. Predicting Experimental Success in De Novo Binder Design: A Meta-Analysis of 3,766 Experimentally Characterised Binders, August 2025.

[25] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, January 2018. arXiv:1706.08500 [cs].

[26] Hannes Stark, Felix Faltings, MinGyu Choi, Yuxin Xie, Eunsu Hur, Timothy O'Donnell, Anton Bushuiev, Talip Uçar, Saro Passaro, Weian Mao, Mateo Reveiz, Roman Bushuiev, Tomáš Pluskal, Josef Sivic, Karsten Kreis, Arash Vahdat, Shamayeeta Ray, Jonathan T. Goldstein, Andrew Savinov, Jacob A. Hambalek, Anshika Gupta, Diego A. Taquiri-Diaz, Yaotian Zhang, A. Katherine Hatstat, Angelika Arada, Nam Hyeong Kim, Ethel Tackie-Yarboi, Dylan Boselli, Lee Schnaider, Chang C. Liu, Gene-Wei Li, Denes Hnisz, David M. Sabatini, William F. DeGrado, Jeremy Wohlwend, Gabriele Corso, Regina Barzilay, and Tommi Jaakkola. BoltzGen: Toward Universal Binder Design, November 2025.

[27] Yehlin Cho, Griffin Rangel, Gaurav Bhardwaj, and Sergey Ovchinnikov. Protein Hunter: exploiting structure hallucination within diffusion for protein design, October 2025.

[28] Tianyu Lu, Richard Shuai, Petr Kouba, Zhaoyang Li, Yilin Chen, Akio Shirali, Jinho Kim, and Po-Ssu Huang. Conditional Protein Structure Generation with Protpardelle-1c, August 2025.

[29] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, November 2017.

[30] William Peebles and Saining Xie. Scalable Diffusion Models with Transformers, March 2023. arXiv:2212.09748 [cs].

[31] J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, A. Courbet, R. J. De Haas, N. Bethel, P. J. Y. Leung, T. F. Huddy, S. Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. K. Bera, N. P. King, and D. Baker. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 378(6615):49–56, October 2022.

[32] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large Language Diffusion Models, February 2025. arXiv:2502.09992 [cs].

[33] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models, December 2020. arXiv:2006.11239 [cs].

[34] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N. Kinch, R. Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R. Glassman, Andy DeGiovanni, Jose H. Pereira, Andria V. Rodrigues, Alberdina A. Van Dijk, Ana C. Ebrecht, Diederik J. Opperman, Theo Sagmeister, Christoph Buhlheller, Tea Pavkov-Keller, Manoj K. Rathinaswamy, Udit Dalwadi, Calvin K. Yip, John E. Burke, K. Christopher Garcia, Nick V. Grishin, Paul D. Adams, Randy J. Read, and David Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, August 2021.

[35] Justas Dauparas, Gyu Rie Lee, Robert Pecoraro, Linna An, Ivan Anishchenko, Cameron Glasscock, and David Baker. Atomic context-conditioned protein sequence design using LigandMPNN. *Nature Methods*, 22(4):717–723, April 2025.

[36] Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishub Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with AlphaFold-Multimer, October 2021.

[37] Casper A. Goverde, Martin Pacesa, Nicolas Goldbach, Lars J. Dornfeld, Petra E. M. Balbi, Sandrine Georgeon, Stéphane Rosset, Srajan Kapoor, Jagrity Choudhury, Justas Dauparas, Christian Schellhaas, Simon Kozlov, David Baker, Sergey Ovchinnikov, Alex J. Vecchio, and Bruno E. Correia. Computational design of soluble and functional membrane protein analogues. *Nature*, 631(8020):449–458, July 2024.

Appendix

A Flow Matching

A.1 Setup

Flow matching involves two main steps: (i) defining a probability path p_t interpolating between a source distribution p, such as a Gaussian, and the target data distribution q, and (ii) training a neural network u_t^θ to regress onto a vector field u_t that generates this path.

For a datapoint $x_1 \sim q$, a common conditional probability path is

$$p_t(x \mid x_1) = \mathcal{N}\big(x \mid tx_1, (1-t)^2 I_d\big).$$

Equivalently, we can sample from this distribution as

$$X_t = tX_1 + (1-t)X_0, \quad X_0 \sim \mathcal{N}(0, I_d).$$

Marginalizing over x_1 gives the probability path

$$p_t(x) = \int p_t(x \mid x_1) q(x_1)\, dx_1.$$

In practice, we sample X_t by drawing $X_1 \sim q$ and $X_0 \sim p$, then forming X_t as above.

The original flow matching loss regresses u_t^θ onto the marginal vector field u_t:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t, X_t \sim p_t} \, \|u_t^\theta(X_t) - u_t(X_t)\|^2,$$

but computing u_t is generally intractable. Lipman et al. [15] showed that regressing onto the conditional vector field yields the same gradients as the intractable marginal version, while being easy to compute. In practice, we therefore use the conditional flow matching (CFM) loss:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t, X_0, X_1} \, \|u_t^\theta(X_t) - u_t(X_t \mid X_1)\|^2,$$

with $X_t = tX_1 + (1-t)X_0$. For the linear path, the conditional velocity field is

$$u_t(x \mid x_1) = \frac{x_1 - x}{1-t},$$

which, when evaluated at X_t, simplifies to $u_t(X_t \mid X_1) = X_1 - X_0$. This leads to the simple training objective:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t, X_0, X_1} \, \|u_t^\theta(tX_1 + (1-t)X_0) - (X_1 - X_0)\|^2.$$

A.2 Classifier-Free Guidance

While flow matching enables unconditional generation, many applications require conditional generation. In T2I, the goal is to generate images matching a text prompt; similarly, we aim to generate binders conditioned on a target protein sequence "prompt".

Classifier-free guidance (CFG) [12] provides a simple mechanism for conditional generation. The idea is to train a single network $u_t^\theta(x \mid y)$ that can operate in both conditional and unconditional modes, where y denotes the conditioning input (e.g., a text prompt or a protein embedding). Crucially, the data distribution consists of *paired samples* $(X_1, y) \sim p_{\text{data}}$, such as an image and its caption, or a binder and its target protein. During training, the conditioning y is randomly replaced with a null token $\varnothing$ with probability η (commonly 10%). This forces the network to learn both $u_t^\theta(x \mid y)$ and $u_t^\theta(x \mid \varnothing)$ with shared parameters.

Formally, the conditional flow matching (CFM) objective under CFG becomes

$$\mathcal{L}_{\text{CFM}}^{\text{CFG}}(\theta) = \mathbb{E}_{t \sim \mathcal{U}[0,1],\, (X_1, y) \sim p_{\text{data}},\, X_0 \sim p_0} \, \|u_t^\theta(X_t \mid y') - u_t(X_t \mid X_1)\|^2,$$

where

$$X_t = tX_1 + (1-t)X_0, \quad y' \leftarrow \begin{cases} y & \text{with prob. } 1-\eta, \\ \varnothing & \text{with prob. } \eta. \end{cases}$$

At inference time, generation is guided by interpolating between unconditional and conditional predictions:

$$\tilde{u}_t(x \mid y) = u_t^\theta(x \mid \varnothing) + \omega\big(u_t^\theta(x \mid y) - u_t^\theta(x \mid \varnothing)\big),$$

where ω is referred to as the the guidance scale. $\omega = 1$ recovers the conditional model as trained, $\omega = 0$ ignores the conditioning, and larger values bias samples more strongly toward the conditioning signal.

Embedding-based conditioning. T2I models often represent y as a continuous text embedding (e.g., BERT, T5, CLIP [16]) rather than a discrete label, enabling generalization to arbitrary and unseen prompts by capturing semantic relationships beyond a fixed vocabulary.

Analogously, in protein binder design, we can construct a dataset of paired examples (X_1, y), where X_1 is a binder and y is a representation of its target protein. Instead of assigning each target protein a discrete token, we can use an embedding obtained from a pretrained protein language model (pLM). In theory, this enables generalization to unseen targets, by leveraging the semantic similarities captured by the pLM embeddings. In this work, we use compressed representations from Raygun [13]. Rather than a discrete null token, we multiply the conditioning by 0.

B Training and Inference

B.1 Dataset

We use the pre-processed STRING dataset released by Ullanat et al. for training MINT [9]. Ullnat et al. start from 2.4 billion of protein–protein interactions (PPIs), spanning 59.3 million unique sequences, protein sequences were clustered at 50% sequence identity using `mmseqs2` [29], and each pair of clusters was represented by at most one interaction to avoid redundancy. After this sequence-identity based de-duplication step, 382 million PPIs from 29 million sequences remained. 250k PPIs were set aside to construct the validation set. To reduce leakage, any clusters from the training set are removed, if the cluster is present in the validation set, resulting in ∼95.8M interactions for training and 250k for validation.

Starting with this processed train/validation split, we reduce the number of total training points by only including interactions with a STRING score above 800 and both sequences are longer than 50 AAs long. This results in 5,192,190 interactions for training, comprising of 2,812,641 unique sequences. We do the same for the validation set, resulting in 248,536 interactions from 439,018 unique sequences.

For our model, each interacting pair (A, B) is encoded into the Raygun latent space as (A', B'), with B' serving as the conditioning input and A' as the datapoint X_1. We train with both orientations, i.e. (A, B) and (B, A), effectively doubling the size of conditioning pairs.

B.2 Model Architecture

We use a Flag-DiT [11] backbone (Figure 2) adapted for protein embeddings, with input and hidden dimension of 1280, 20 layers, and 20 attention heads per layer. We arbitrarily chose these values as a reasonable starting point and did not test any other configuration for the prototype. Each layer consists of multi-head self-attention and an MLP block with feedforward ratio of 2.0. While a ratio of 4.0 is standard, we halved the ratio to fit the model in one H100 with 80Gb of memory.

B.3 Training Details

The model is trained with the CFM loss with classifier-free guidance dropout ($\eta = 0.1$). Training ran for 500k optimization steps on 2 NVIDIA H100 GPUs using Distributed Data Parallel, with a per-GPU batch size of 512. We optimized with AdamW with default parameters and a constant learning rate of 1×10^{-4}. Per common practice, an exponential moving average (EMA) with decay 0.9999 was maintained. Following Peebles et al. [30] who suggested DiTs need minimal training heuristics, we used no weight decay, gradient clipping, or learning rate warm-up. Validation was performed on held-out protein pairs, with the lowest validation CFM loss used for checkpoint selection. To generate Raygun embeddings, we use the latest `raygun_8_8mil_800M` checkpoint.

B.4 Inference

To generate binders for a given target, we sample noise $X_0 \sim \mathcal{N}(0, I_d)$ in the Raygun latent space and integrate the learned ODE

$$\frac{dX_t}{dt} = u_t^\theta(X_t, t)$$

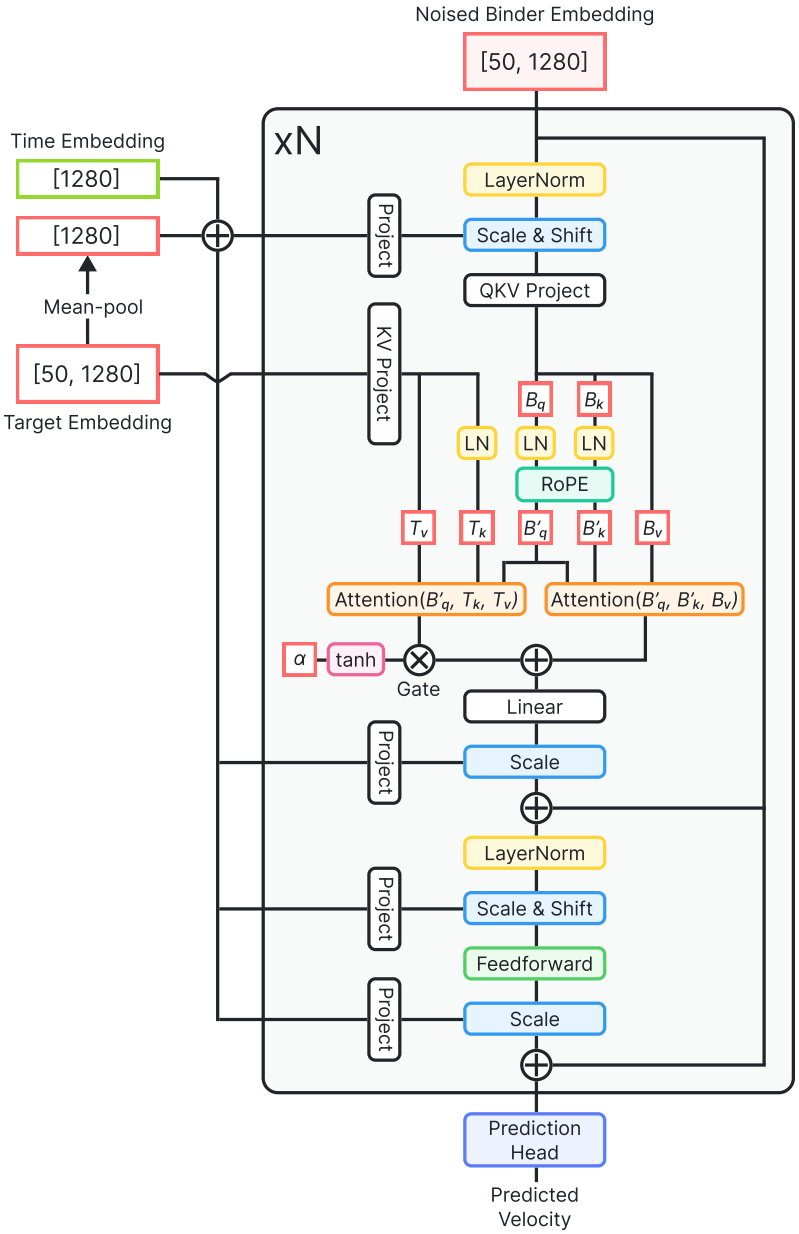

Figure 2: **Flag-DiT-T2I Block.** We use LayerNorm instead of the originally proposed RMSNorm to simplify the prototype. We use the same time embedding module used by the original DiT paper.

from $t = 0$ to $t = 1$. We use 100 Euler steps with a guidance scale of 1.0, producing a binder embedding X_1 that is decoded into an amino acid sequence by Raygun. The Euler update rule is

$$X_{t+\Delta t} = X_t + \Delta t\, u_t^\theta(X_t, t), \quad \Delta t = 1/100.$$

The use of higher-order solvers remains to be explored.

The generated embedding is projected back into $\mathbb{R}^{L\times 1280}$ ESM2 space by Raygun, where L is the desired sequence length (here $L = 50$ for all minibinder experiments). Raygun's pre-trained ESM2 embedding decoder then maps these embeddings to logits, from which sequences are obtained via greedy decoding. Special tokens are prevented from being selected, including the unknown residue `X`. In the future, we can also prevent cysteine, `C`, from being decoded to avoid the formation of disulfide bridges.

C Selected Methods

Binder design methods have grown rapidly, though many remain closed-source [1]. Here, we focus on widely adopted, state-of-the-art open-source approaches. RFDiffusion/ProteinMPNN [2, 31], the most established structure-based pipeline, serves as a natural baseline. Diffusion Sequence Model (DSM) [23], is a recent protein language model trained with masked diffusion. When fine-tuned on pairs of interacting proteins from STRING, DSM is capable of generating binders through un-masking, making it an informative sequence-based comparison.

BindCraft [3] represents an end-to-end binder discovery pipeline rather than a purely generative model. To provide a point of comparison, we focus on its AlphaFold-hallucination trajectory generation step for benchmarking, while acknowledging this removes a core component of the full BindCraft workflow. We note this comparison is not strictly apples-to-apples, as BindCraft trajectories are optimized directly for structure-based metrics, which we also use for evaluation.

C.1 DSM

The Diffusion Sequence Model (DSM) [23] is a protein language model that extends ESM2 with a masked diffusion training framework (Large Language Diffusion Models (LLaDa) [32]). Unlike conventional pLMs, which focus mainly on representation learning, DSM is trained to both reconstruct heavily corrupted sequences and generate new sequences, enabling unified representation and generative modeling.

For binder design, DSM was fine-tuned (DSM-ppi) on a curated set of high-confidence PPIs from STRING. Interacting pairs are concatenated into a single sequence, and during training the partner sequence is masked and reconstructed in the presence of the target sequence. This setup allows DSM to learn context-dependent generation: given a target protein (SeqA), DSM generates a plausible interacting binder (SeqB).

C.1.1 DSM Inference

We use the recommended checkpoint (DSM_ppi_full) with default settings: (`step_divisor`=100, `temperature`=1.0, and `remasking`="random").

C.2 RFDiffusion and ProteinMPNN

RFdiffusion [2] is a generative framework for protein backbone design built on denoising diffusion probabilistic models (DDPMs) [33]. By fine-tuning the RoseTTAFold [34] structure prediction network on denoising tasks, RFdiffusion learns to generate diverse and realistic protein backbones while supporting conditioning on structural or functional motifs. This conditioning enables applications ranging from enzyme active site scaffolding to de novo binder design against specific protein targets.

In binder design, RFdiffusion generates backbone structures compatible with a target protein surface. These backbones can then be mapped to the most likely amino acid sequence with ProteinMPNN [31], an inverse folding model, or more formally a structure-conditioned sequence generation model. Together, RFdiffusion and ProteinMPNN form a widely used two-stage binder design pipeline: backbone generation via diffusion, followed by sequence design. This method has been experimentally

validated across tasks such as symmetric oligomer design, therapeutic scaffold design, and binder design.

C.2.1 RFDiffusion/ProteinMPNN Inference

For RFDiffusion, we used the default inference code:

```
./scripts/run_inference.py inference.output_prefix=generated_sequences/TARGET
NAME inference.input_pdb=input_pdbs/TARGET PDB.pdb 'contigmap.contigs=[TARGET
CONTIG/0 50-50]' inference.num_designs=100 denoiser.noise_scale_ca=0
denoiser.noise_scale_frame=0
```

Where the `TARGET NAME`, `TARGET PDB`, and `TARGET CONTIG` refer to the corresponding target information from Table 3.

For ProteinMPNN, we ran the default inference code from the LigandMPNN [35] repository.

```
run.py -seed $i -pdb_path "[RFDiffusion OUTPUT].pdb" -out_folder
"./generated_sequences/[TARGET NAME]" -chains_to_design "A" -temperature
0.1
```

Where `[RFDiffusion OUTPUT]` refers to the name of generated backbone PDB file from the previous step, and `TARGET NAME` refers to the current target we are generating binders for.

C.3 BindCraft

BindCraft [3] is a binder design pipeline based on hallucination through AlphaFold2 [19]. Unlike structure-based diffusion approaches such as RFdiffusion, which keep the target backbone fixed, BindCraft uses AlphaFold to repredict the binder–target complex at each iteration, allowing both binder and target backbones to adapt to the interface. Binder sequences are optimized by back-propagating error gradients through AF2 weights, updating sequence and structure simultaneously.

In the original BindCraft pipeline, initial designs are generated with AF2-Multimer [36] and then further refined with Soluble-MPNN [37] to improve core and surface sequences while preserving the designed interface. Final candidates are re-predicted with AF2-Monomer to reduce bias toward multimer training and filtered with AF2 confidence metrics and Rosetta-based energy scores.

C.3.1 BindCraft Inference

As discussed above, we attempted to separate the role of the generative stage from the filtering step. As a result, we evaluate the trajectory generation step of BindCraft, rather than the full pipeline. For each target, we generated 100 trajectories using a modified configuration of the BindCraft pipeline. We used the `default_4stage_multimer.json`, keeping the original thresholds for advancing design stages. We note that not all trajectories pass through all four design stages but are still included as part of the filter-free evaluation.

We specified the chain from the PDB files in Table 3 without specifying hotspot residues.

D ipSAE

The interaction prediction score from aligned errors (ipSAE) was proposed as a more robust alternative to AlphaFold's interface predicted template modeling score (ipTM) [36]. Unlike ipTM, which is sensitive to non-interacting regions and disordered domains when full-length sequences are used, ipSAE restricts evaluation to interchain residue pairs with confident predicted aligned error (PAE) values. This makes it more reliable in realistic settings where accessory domains are common.

ipSAE is asymmetric, meaning the score between binder-to-target may be different from target-to-binder. *ipSAE_max* takes the higher value of the two, whereas *ipSAE_min* takes the smaller. It is reasoned that *ipSAE_min* captures the "weakest link" across chains and provides a more stringent evaluation. As the meta-analysis by Overath et al. show ipSAE_min is more discriminative than ipSAE_max [24], we calculate and report ipSAE_min using the default PAE and distance cutoff of 10 for all experiments using the outputs of Boltz-2.

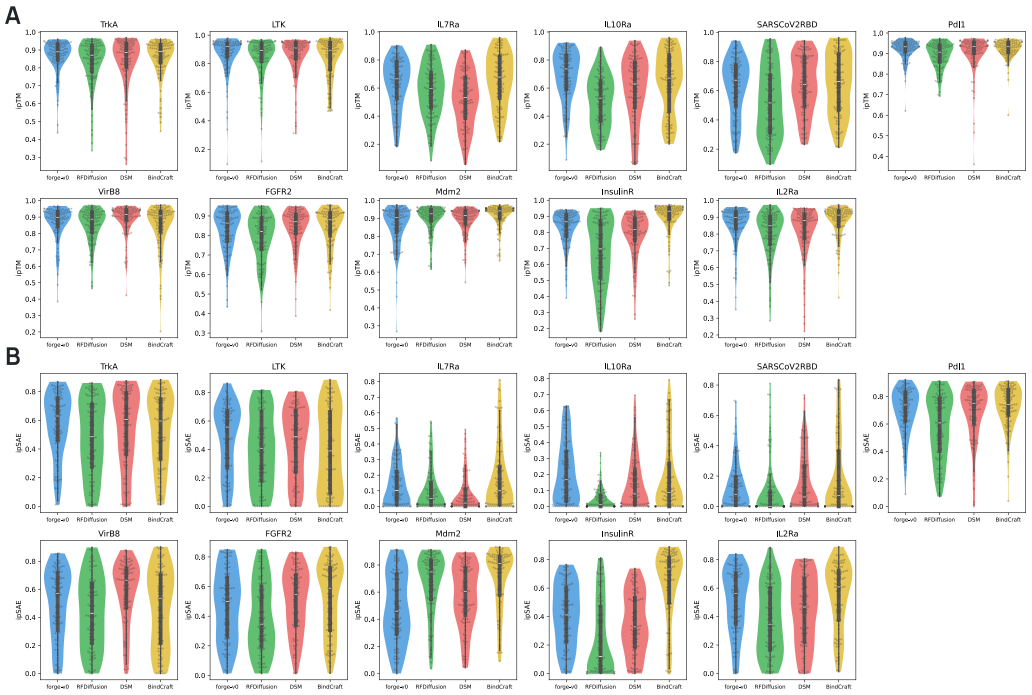

Figure 3: **Distribution of ipTM (A) and ipSAE (B) scores per target and method.** Not all points are visualized on the swarmplot due to size.

E Additional Experiments

E.1 Distribution of ipTM and ipSAE scores

For additional context, we visualize the distribution of ipTM and ipSAE scores per target per model as a combined violin/swarm plot (Figure 3).

E.2 The Amino Acid Distribution of `forge-v0` is Distinct from DSM

Given that both DSM and `forge-v0` are sequence-based models trained on a subset of STRING, a potential concern was `forge-v0` is redundant, and mirrors the behavior of DSM. We looked to further understand their differences by analyzing the binder residues at the predicted interfaces.

To analyze the residue composition of binders at the interface, we select the ten binders with the highest ipSAE score per method, per target and define interface residues as binder residues whose α-carbon is within 8Å of any target α-carbon in the predicted complex structure. We focus on the top 10% of designs rather than all generated binders, as this highlights the most promising candidates and provides a fairer comparison of how each model performs under its best-case scenarios.

Comparing the two interface residue distributions, we observe a clear distinction between the two models (Figure 4). `forge-v0` shows higher usage of polar residues such as threonine (T) and glutamine (Q), whereas DSM produces more proline (P) and glycine (G). These differences indicate `forge-v0` learns a distinct generative bias, motivating further development and differentiation from DSM.

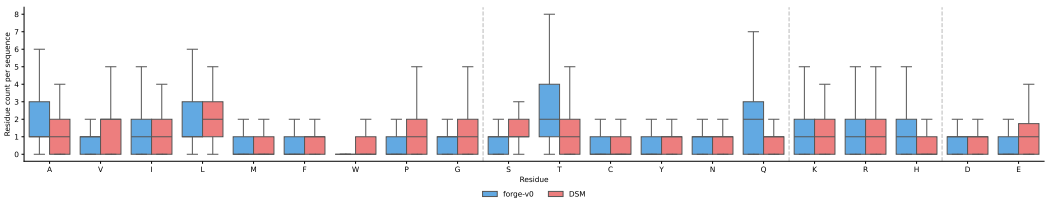

Figure 4: Interface residue composition for DSM and `forge-v0` generated sequences.

Target	PDB ID	Chain	Contig (Residues)	Sequence
VirB8	4o3v	A	90–206, 211–231	ANPYISVANIMLQNYVKQREKYNYDTLKEQFTFIKNA STSIVYMQFANFMNIDNSLSPVIRYQKLYRRSINIISIN NINNNEATVTFESLAQNNTGEILENMLWEAKIGFIMD SISTNMPFHFIVTSYKLKLLRNKNQ
InsulinR	4oga	E	6–155	EVCPGMDIRNNLTRLHELENCSVIEGHLQILLMFKTRP EDFRDLSFPKLIMITDYLLLFRVYGLESLKDLFPNLTVI RGSRLFFNYALVIFEMVHLKELGLYNLMNITRGSVRI EKNNELCYLATIDWSRILDSVEDNHIVLNKDDNEEC
TrkA	2ifg	A	282–382	VSFPASVQLHTAVEMHHWCIPFSVDGQPAPSLRWLF NGSVLNETSFIFTEFLEPAANETVRHGCLRLNQPTHVN NGNYTLLAANPFGQASASIMAAFMDNP
FGFR2	1djs	A	251–296, 307–362	RSPHRPILQAGLPANASTVVGGDVEFVCKVYSDAQPH IQWIKHVPYLKVLKAAGVNTTDKEIEVLYIRNVTFED AGEYTCLAGNSIGISFHSAWLTVLPAP
IL7Ra	3di3	B	17–209	DYSFSCYSQLEVNGSQHSLTCAFEDPDVNTTNLEFEIC GALVEVKCLNFRKLQEIYFIETKKFLLIGKSNICVKVG EKSLTCKKIDLTTIVKPEAPFDLSVVYREGANDFVVTF NTSHLQKKYVKVLMHDVAYRQEKDENKWTHVNLSS TKLTLLQRKLQPAAMYEIKVRSIPDHYFKGFWSEWSP SYYFRTP
SARS-CoV-2 RBD	6m0j	E	333–526	TNLCPFGEVFNATRFASVYAWNRKRISNCVADYSVL YNSASFSTFKCYGVSPTKLNDLCFTNVYADSFVIRGD EVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSNNL DSKVGGNYNYLYRLFRKSNLKPFERDISTEIYQAGSTP CNGVEGFNCYFPLQSYGFQPTNGVGYQPYRVVVLSF ELLHAPATVCGP
PD-L1	7uxo	A	18–131	NAFTVTVPKDLYVVEYGSNMTIECKFPVEKQLDLAA LIVYWEMEDKNIIQFVHGEEDLKVQHSSYRQRARLLK DQLSLGNAALQITDVKLQDAGVYRCMISYGGADYKR ITVKVN
MDM2	1ycr	A	25–109	ETLVRPKPLLLKLLKSVGAQKDTYTMKEVLFYLGQYI MTKRLYDEKQQHIVYCSNDLLGDLFGVPSFSVKEHR KIYTMIYRNLVV
LTK	7nx0	C	66–190, 202–244, 251–378	GSWLFSTCGASGRHGPTQTQCDGAYAGTSVVVTVGA AGQLRGVQLWRVPGPGQYLISAYGAAGGKGAKNHL SRAHGVFVSAIFSLGLGESLYILVGQQGEDACPGGSPE SQLVCLGESRAVEEHAAMARWAGGGGGGGGATYVF RVRAGELEPLLVAAGGGGRAYLRPRDRGASPEKLEN RSEAPGSGGRGGAAGGGGGWTSRAPSPQAGRSLQEG AEGGQGCSEAWATLGWAAAGGFGGGGGACTAGGG GGGYRGGDASETDNLWADGEDGVSFIHPSSELFLQPL AVTENHGEVEIRRH
IL10Ra	1lqs	R	2–208	GTELPSPPSVWFEAEFFHHILHWTPIPQQSESTCYEVA LLRYGIESWNSISQCSQTLSYDLTAVTLDLYHSNGYR ARVRAVDGSRHSQWTVTNTRFSVDEVTLTVGSVNLE IHNGFILGKIQLPRPKMAPAQDTYESIFSHFREYEIAIR KVPGQFTFTHKKVKHEQFSLLTSGEVGEFCVQVKPSV ASRSNKGMWSKEECISLTRQ
IL2Ra	1z92	B	1–47, 53–64, 104–165	ELCDDDPPEIPHATFKAMAYKEGTMLNCECKRGFRRI KSGSLYMLCTGNSSHSSWDNQCQCTSSATRNTTKQV TPQPEEQKERKTTEMQSPMQPVDQASLPGHCREPPPW ENEATERIYHFVVGQMVYYQCVQGYRALHRGPAESV CKMTHGKTRWTQPQLICTG

Table 3: Target PDB IDs, Chain IDs, Contig residue ranges, and sequences used. Note, contig residues are sometimes gapped (e.g. 1z92) due to missing residues in the structure. These 11 targets are chosen from the 15 analyzed by Overath et al. to represent common targets. We excluded EGFR and other targets with multiple epitopes to simplify benchmarking at this stage.