
Learning the Language of Protein Structure

Benoit Gaujac*

InstaDeep

b.gaujac@instadeep.com

J r mie Don *

InstaDeep

j.dona@instadeep.com

Liviu Copoiu

InstaDeep

l.copoiu@instadeep.com

Timothy Atkinson

InstaDeep

t.atkinson@instadeep.com

Thomas Pierrot

InstaDeep

t.pierrot@instadeep.com

Thomas D. Barrett

InstaDeep

t.barrett@instadeep.com

Abstract

Representation learning and *de novo* generation of proteins are pivotal computational biology tasks. Whilst natural language processing techniques have proven highly effective for protein sequence modelling, structure modelling presents a complex challenge, primarily due to its continuous and three-dimensional nature. Motivated by this discrepancy, we introduce an approach using a vector-quantized autoencoder that effectively tokenizes protein structures into discrete representations. This method transforms the continuous, complex space of protein structures into a manageable, discrete format with a codebook ranging from 4096 to 64000 tokens, achieving high-fidelity reconstructions with backbone root mean square deviations (RMSD) of approximately 1-5  . To demonstrate the efficacy of our learned representations, we show that a simple GPT model trained on our codebooks can generate novel, diverse, and designable protein structures. Our approach not only provides representations of structures, but also mitigates the challenges of disparate modal representations and sets a foundation for seamless, multi-modal integration, enhancing the capabilities of computational methods in protein design.

1 Introduction

The application of machine learning to large-scale biological data has ushered in a transformative era in computational biology, advancing both representation learning and *de novo* generation. The integration of machine learning in molecular biology has led to significant breakthroughs [Sapoval et al., 2022, Chandra et al., 2023, Khakzad et al., 2023, J nes and Beltrao, 2024], spanning diverse and complex data modalities such as sequences, structures, functional descriptors.

The deep learning landscape is increasingly centered around attention-based architectures, particularly transformers [Vaswani et al., 2017], due to their performance and scalability. Transformers have proven effective across various domains, including representation [Radford et al., 2021], image and audio generation [Chen et al., 2020, Chang et al., 2022, Ziv et al., 2024], and reinforcement learning [Chen et al., 2021, Boige et al., 2023]. Large multi-modal models (LMMs) leveraging transformer backbones are emerging as key tools in ubiquitous AI (GPT4 [Achiam et al., 2023], LLaVA [Liu et al., 2024], Gemini [Gemini et al., 2023], Flamingo [Alayrac et al., 2022]), text-conditioned generation of images (Parti [Yu et al., 2022b], Muse [Chang et al., 2023]) and sounds (MusicGen [Copet et al., 2023]), and even specialized fields like medicine (Med-Gemini [Yang et al., 2024b], Med-PaLM [Tu et al., 2024]) and genomics (ChatNT [Richard et al., 2024]). These LMMs

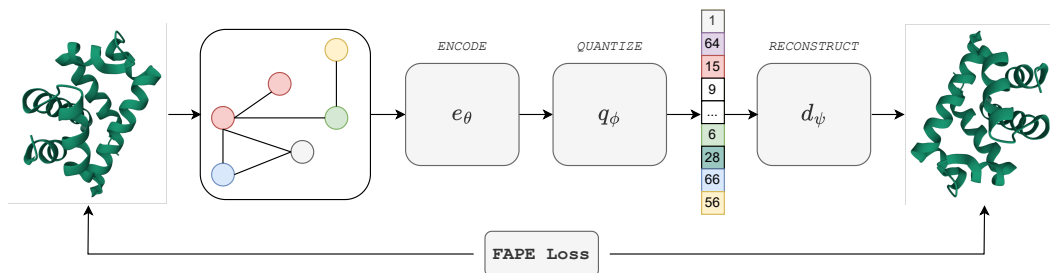


Figure 1: Schematic overview of our approach. The protein structure is encoded as a graph to extract features from using a GNN. This embedding is quantized before being fed to the decoder to estimate the positions of backbone atoms.

use pre-trained encoders to combine modalities in sequence space. However, a methodology for applying sequence modeling to protein structures is yet to be established. Foldseek [van Kempen et al., 2024] proposes a quantized autoencoder to encode protein geometry, successful for database search and augmenting residue-based vocabularies in LMMs [Su et al., 2024]. However, it only extracts local features, preventing its use for tasks like structure generation or binding prediction that require global information [Krapp et al., 2023]. Structure-based modeling of proteins remains challenging due to their 3D and continuous nature, complicating the direct application of transformer models designed for discrete data. Instead, bespoke geometric deep learning methodologies are employed, such as GNN [Dauparas et al., 2022, Krapp et al., 2023] and structure-aware modules as in AlphaFold [Jumper et al., 2021]. Generative modeling of structures typically uses methods designed for continuous variables, such as diffusion [Watson et al., 2023, Yim et al., 2023] and flow matching [Bose et al., 2024], rather than discrete-variable models successful in sequence modeling.

To address this gap, we learn quantized representations of protein structures to leverage sequence-based language models. Our objectives are: (i) **To encode protein structures in a discrete latent space:** We propose transforming structural information of proteins into discrete sequential data, enabling seamless integration with sequence-based models. (ii) **To achieve a low reconstruction error:** We aim to minimize the reconstruction error typically reaching the range of 1-5 Ångströms. Our contributions are threefold. First, we introduce quantized autoencoders that discretize protein structures into sequences of tokens while preserving information for accurate reconstruction. Second, we validate our autoencoders through qualitative and quantitative analysis, and various ablation studies. Third, we demonstrate the practicality of the learned representations by training a simple GPT model on our codebook, which successfully generates novel, diverse, and structurally viable protein structures. All source code is publicly available online (<https://github.com/instadeepai/protein-structure-tokenizer/>).

2 Method

2.1 Protein Structure Autoencoder

Our objective is to train an autoencoder that maps protein structures to and from a discrete latent space of sequential codes. Following prior works [Yim et al., 2023, Wu et al., 2024], we consider the backbone atoms of a protein, $N-C_\alpha-C-O$, to define the overall structure. For a protein consisting of N residues, we seek to map its structure, represented by the tensor of the backbone atoms coordinates $\mathbf{p} \in \mathbb{R}^{N \times 4 \times 3}$, to a latent representation $\tilde{\mathbf{z}} = [\tilde{\mathbf{z}}_1, \dots, \tilde{\mathbf{z}}_{\frac{N}{r}}]$, where a r is a downsampling ratio, controlling the size of the representation. Note that each element $\tilde{\mathbf{z}}_i$ can only take a finite number of values, with the collection of all possible values defining a codebook \mathcal{C} . A schematic overview of the our autoencoder is depicted in Figure 1. In this section, we focus on: the *encoder* e_θ extracting a set of $\frac{N}{r}$ embeddings of dimension c denoted $\mathbf{z} \in \mathbb{R}^{\frac{N}{r} \times c}$, the *quantizer* q_ϕ that discretizes \mathbf{z} to obtain a quantized representation $\tilde{\mathbf{z}}$, and the *decoder* d_ψ that predicts a structure $\tilde{\mathbf{p}} \in \mathbb{R}^{N \times 4 \times 3}$ from $\tilde{\mathbf{z}}$. The learnable parameters are respectively denoted (θ, ϕ, ψ) and the learning setting summarizes as:

$$\mathbf{p} \xrightarrow{e_\theta} \mathbf{z} \xrightarrow{q_\phi} \tilde{\mathbf{z}} \xrightarrow{d_\psi} \tilde{\mathbf{p}}.$$

Encoder The encoder maps the backbone atoms positions $\mathbf{p} \in \mathbb{R}^{N \times 4 \times 3}$, to a continuous downsampled continuous representation $\mathbf{z} \in \mathbb{R}^{\frac{N}{r} \times c}$ where r is the downsampling ratio:

$$e_\theta : \mathbf{p} \in \mathbb{R}^{N \times 4 \times 3} \mapsto \mathbf{z} \in \mathbb{R}^{\frac{N}{r} \times c} \quad (1)$$

Note that when the downsampling ratio r is set to 1, each component $\mathbf{z}_i \in \mathbb{R}^c$ can be interpreted as the encoding of residue i . This representation learning task is similar to mapping point-clouds to sequences [Yang et al., 2024a, Boget et al., 2024]. Inverse folding [Ingraham et al., 2019, Dauparas et al., 2022], which estimates amino acid sequences from protein structures, is another example. ProteinMPNN has shown remarkable success in latter task. Following their design, we parameterize our encoder using a Message-Passing Neural Network (MPNN) [Dauparas et al., 2022]. Additionally, we introduce an attention mechanism, detailed in Appendix A.2 and Algorithm 2, to compress the structure representation by a downsampling ratio r . The encoding scheme is detailed in Appendix A.3.

Quantization The quantizer plays a crucial role by discretizing continuous latent representations into discrete codes. Instead of employing traditional direct codebook learning methods [van den Oord et al., 2017, Razavi et al., 2019], we adopt the Finite Scalar Quantization (FSQ) framework [Mentzer et al., 2024], which we found to be more stable during training. FSQ learns a discrete latent space by rounding a bounded low-dimensional encoding of the latent representation. Consider $\mathbf{z}_i \in \mathbb{R}^c$, the i^{th} element of the continuous encodings \mathbf{z} , and the quantization levels $\mathbf{L} = [L_1, \dots, L_d] \in \mathbb{N}^d$. Its discretized counterpart, denoted as $\tilde{\mathbf{z}}_i \in \mathbb{Z}^d$, typically with $d \leq 8$, is defined by:

$$\tilde{\mathbf{z}}_i = \text{round} \left(\frac{\mathbf{L}}{2} \odot \tanh(W \mathbf{z}_i) \right) \quad (2)$$

where \odot the element-wise multiplication and $W \in \mathbb{R}^{d \times c}$ is a weight matrix (details in Appendix A.2). In doing so, each quantized vector $\tilde{\mathbf{z}}_i$ can be mapped to an index in $\{1, \dots, \prod_{j=1}^d L_j\}$. This defines a codebook by its indices, associating an index to a unique combination of the each dimension values.

Decoder The decoder is tasked with estimating a structure $\hat{\mathbf{p}}$ from the quantized representation $\tilde{\mathbf{z}}$:

$$d_\psi : \tilde{\mathbf{z}} \in \mathbb{R}^{\frac{N}{r} \times c} \mapsto \hat{\mathbf{p}} \in \mathbb{R}^{N \times 4 \times 3} \quad (3)$$

The task our decoder addresses formulates as a sequence to point-cloud task. A paradigmatic example of such a task in biology is the protein folding problem, where a protein’s conformation is estimated given its sequence of amino-acids. We second Jumper et al. [2021] who tackled this task proposing a novel architecture for point cloud estimation from latent embeddings, parameterizing a point cloud using *frames* defined by a tuple $\mathbf{T} = (\mathbf{R}, \mathbf{t})$ where \mathbf{R} is the frames’ orientation and \mathbf{t} is the frame center. The origin of the frame is set to the C_α carbon, and the orientation is defined using the nitrogen and the other carbon atom, see details in Jumper et al. [2021] and Algorithm 3 and Appendix A.2.

Objective For training, we resort to the Frame Align Point Error (FAPE) loss, introduced in Jumper et al. [2021], enabling to compare two sets of point clouds. Given a ground-truth frame \mathbf{T}_i and ground-truth atom position \mathbf{x}_j expressed in \mathbf{T}_i , and their predictions \mathbf{T}_i^P and \mathbf{x}_j^P , the FAPE Loss is:

$$\mathcal{L}_{\text{FAPE}} = \|\mathbf{T}_i^{P^{-1}}(\mathbf{x}_j^P) - \mathbf{T}_i^{-1}(\mathbf{x}_j)\| \quad (4)$$

We train our algorithm on around 310000 structures extracted from the PDB dataset [Berman et al., 2000]. We defer datasets creation details and model hyperparameters to Appendix A.3.

3 Experiments

In this section we first evaluate, both qualitatively and quantitatively, our vector-quantized autoencoder by considering the compression and reconstruction performance and demonstrate that our tokenizer indeed permits high-accuracy reconstruction of protein sequences. We then further highlight how this can be adapted to downstream tasks, by effectively training a *de novo* generative model for protein structures using a vanilla decoder-only transformer model trained on the next token prediction task.

3.1 Autoencoder Evaluation

Experiments We trained several versions of the quantized autoencoder; with small ($K = 4096$ codes) and large ($K = 64000$) codebooks and increasing downsampling ratio (r) (from 1 to 4). For the downsampling ratio of 1, we also trained codebooks with 432 and 1728 codes to further show the compression ability of our algorithms. For each codebook we evaluate the reconstruction performance achieved on the held out test set, to understand the trade-offs between compression and expressivity associated with these hyperparameter choices.

Metrics To assess the reconstruction performances of the models, we rely on standard structural biology metrics: The *root mean square distance (RMSD)* and the *TM-score* [Y. and J., 2005]. For context, two structures are considered to have similar fold when their TM-score exceeds 0.5 [Xu and Y., 2010] and a RMSD below 2Å is usually seen as approaching experimental resolution. We report the metrics on two datasets: (1) left-out clusters of proteins (independent from the ones in the train set) and (2) CASP15, a structural dataset recognized for containing complex structures.

Results Our results are summarised in Table 1. We find that with a codebook of $K = 64000$ and a downsampling of $r = 1$; our average reconstruction has 1.59 Å RMSD and a TM-score of 0.95. For comparison, we also report the reconstruction performance of the exact same models without latent quantization. Whilst increasing the model capacity may allow these scores to be improved even further, this is already approaching the limit of experimental resolution (which is to say, the reconstruction errors are on par with the experimental errors in resolving the structures).

Table 1: Average Test set reconstruction results of our discrete auto-encoding method for several down-sampling ratios and (implicit) codebook sizes. For CASP-15 we report the median of the metrics due to the limited dataset size. Note that a RMSD below 2Å is considered of the order of experimental resolution and two proteins with a TM-score > 0.5 are considered to have the same fold. The compression is defined as: the number of bits necessary to store the $N \times 4 \times 3$ backbone positions divided by the number of bits necessary to store the $\frac{N}{r}$ tokens multiplied by $\log_2(\# Codes)$

Downsampling	Number of Codes	Compression Factor	Results		CASP15	
			RMSD (↓)	TM-Score (↑)	RMSD (↓)	TM-Score (↑)
1	432	88	2.09 Å	0.91	1.75 Å	0.89
	1728	71	1.79 Å	0.93	1.33 Å	0.94
	4096	64	1.55 Å	0.94	1.25 Å	0.94
	64000	48	1.22 Å	0.96	0.94 Å	0.97
	without quantization	—	0.97 Å	0.98	1.07 Å	0.93
2	4096	128	2.22 Å	0.90	1.73 Å	0.89
	64000	96	1.95 Å	0.92	1.82 Å	0.90
	without quantization	—	1.45 Å	0.95	1.44 Å	0.90
4	4096	256	4.10 Å	0.81	2.79 Å	0.77
	64000	192	2.96 Å	0.86	2.55 Å	0.80
	without quantization	—	2.19 Å	0.91	1.98 Å	0.84

Table 1 clearly indicates that increasing the downsampling factor or decreasing the codebook size correspondingly impacts the reconstruction accuracy. Nevertheless in all cases we find that the achievable reconstruction performance is still within a few angstrom with TM-scores clearly exceeding the 0.5 threshold on average. Finally, comparing to continuous autoencoders, our learned quantization demonstrates significant information compression at the expense of only a small decrease in the reconstruction precision of the order of 0.5 – 1 Å.

The reconstruction performance is illustrated in Figure 5, where examples of the model’s outputs are superimposed with their corresponding targets in the case of a downsampling factor of $r = 2$ and $K = 64000$ codes. Visually, our model demonstrates its ability to capture the global structure of each protein. Additionally, our model faithfully reconstructs local conformation of each protein preserving essential secondary structures elements of the protein such as the α -helices and β -sheets.

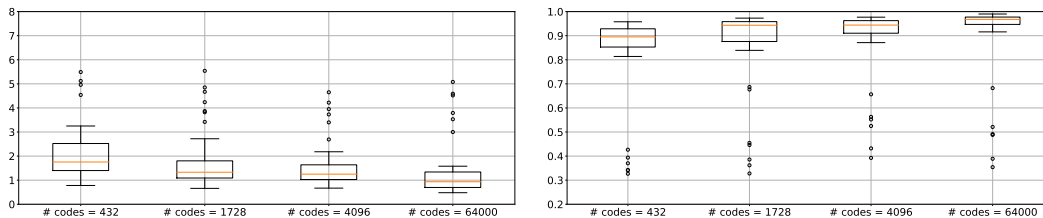


Figure 2: Evolution of the RMSD (left) and TM-score (right) distribution with the codebook size for a downsampling ratio of 1 on CASP-15 data.

This is also noticeable in Figure 2 that shows that, given a downsampling ratio, larger codebooks effectively decrease the reconstruction errors.

3.2 *De novo* protein structure generation

Experiment We now demonstrate that our learned discrete autoencoder can be effectively leveraged for downstream tasks. In particular, we consider generation of protein structures from a model trained in our latent space as a paradigmatic demonstration of our tokenizers utility. This is not just because generative models for *de novo* protein design are of great interest for drug discovery – enabling rapid *in silico* exploration of the design space – but also as it directly leverages our compressed representation of protein structures using established sequence modelling architectures.

Specifically, we tokenize our dataset (full details on dataset preparation for this experiment can be found in Appendix A.4.1) using a downsampling factor of 1 and a codebook with 4096 codes (first line of Table 1). More specifically, we train an out-of-the-box decoder-only transformer model with 20 layers, 16-heads and an embedding dimension of 1024 (344M parameters) on a next-token-prediction task on the training split. This decoder-only model is used to generate new sequences of tokens which are then mapped back to 3D protein structures using our pre-trained decoder.

Metrics We evaluate generated structures on *designability*, *novelty*, and *diversity*. *Designability*, or *self-consistency*, uses ProteinMPNN [Dauparas et al., 2022] to predict sequences for generated structures and refolds them using ESMFold [Lin et al., 2022] to measure structural similarity. We report the proportion of proteins with TM-score (scTM) above 0.5 and RMSD (scRMSD) below 2Å. A useful generative model should produce varied samples, not mere replications. *Diversity* is assessed via structural clustering, reporting the number of clusters normalized by the number of generated sequences. *Novelty* compares generated structures to a reference dataset using TM-score, considering structures novel if their maximum TM-score is below 0.5. We follow Yim et al. [2023] for these metrics and provide more details in Appendix A.4.2.

Baselines We compare our model with recent competitive methods, namely FrameDiff [Yim et al., 2023] and RFDiffusion [Watson et al., 2023]. Both use bespoke SE(3) diffusion models specifically designed and trained the structure generation task. Note that, the state-of-the-art RFDiffusion leverages the extensive pre-training of RoseTTAFold [Baek et al., 2021] on a large dataset and requires considerable computational cost.

Results The results for different metrics are given in Table 2, with sampling strategies described in Appendix A.4.3. Our GPT model generates protein structures comparable to specialized methods like FrameDiff. We report self-consistency and diversity metrics for 1600 randomly sampled structures from our validation in Table 2. While our method shows competitive performance in generating designable domains, the results for novelty and diversity are more nuanced. Our model generates structures closer to the reference dataset, possibly due to sampling parameters favoring modes of the data distribution (Appendix A.4.3). Although generating structures that are novel and diverse is desirable, structures that differ too much from the natural proteins found in the reference dataset, can also indicate unrealistic structures, making the novelty and diversity metrics harder to interpret on their own. Visualizing novelty against designability (Figure 8), we find that 9.01 % of our samples are both novel and designable, compared to 8.18 % for FrameDiff and 58.58 % for RFDiffusion. Additional

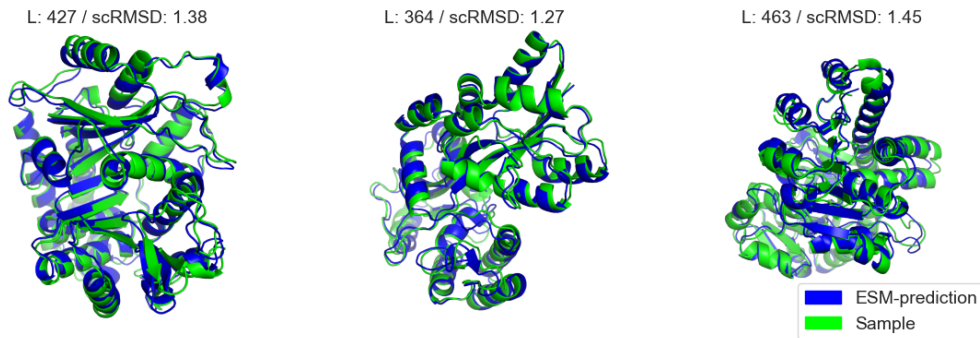


Figure 3: Visualisation of generated samples (green) super-imposed with their self-consistent ESM-predicted structures (blue).

results and analyses are provided in Appendix A.4.4. In Figure 3, we visualize random samples from our model superimposed with ESM-predicted structures, showing diverse structures with non-trivial secondary elements and close alignment with original samples, highlighting designability.

4 Conclusion

This work presents a methodology for learning a discrete representation of protein geometry, mapping structures into sequences of integers while recovering near-native conformations upon decoding. Our primary contribution is enabling the application of standard sequence-modeling techniques to protein structures. The expressiveness of the tokenized representation is crucial for high-fidelity reconstruction of 3D structures. Empirical and visual inspection confirm this for our methodology. A proof-of-concept GPT model trained on tokenized PDB entries demonstrates competitive performance with some recent diffusion-based models. While not yet matching seminal approaches like RFDiffusion, the potential of sequence-modeling algorithms across diverse data modalities suggests a promising direction for future research.

References

- J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv*, 2023.
- J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *Advances in neural information processing systems*, 2022.
- M. Baek, F. DiMaio, I. Anishchenko, J. Dauparas, S. Ovchinnikov, G. R. Lee, J. Wang, Q. Cong, L. N. Kinch, R. D. Schaeffer, C. Millán, H. Park, C. Adams, C. R. Glassman, A. DeGiovanni, J. H. Pereira, A. V. Rodrigues, A. A. van Dijk, A. C. Ebrecht, D. J. Opperman, T. Sagmeister, C. Buhlheller, T. Pavkov-Keller, M. K. Rathinaswamy, U. Dalwadi, C. K. Yip, J. E. Burke, K. C. Garcia, N. V. Grishin, P. D. Adams, R. J. Read, and D. Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 2021.
- H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. Shindyalov, and P. E. Bourne. "The Protein Data Bank". *Nucleic Acids Research*, 2000.
- Y. Boget, M. Gregorova, and A. Kalousis. Discrete graph auto-encoder. *Transactions on Machine Learning Research*, 2024.
- R. Boige, Y. Flet-Berliac, A. Flajolet, G. Richard, and T. PIERROT. PASTA: Pretrained action-state transformer agents. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.
- J. Bose, T. Akhoun-Sadegh, K. Fatras, G. Huguet, J. Rector-Brooks, C.-H. Liu, A. C. Nica, M. Korbalyov, M. M. Bronstein, and A. Tong. SE(3)-stochastic flow matching for protein backbone generation. In *International Conference on Learning Representations*, 2024.

- A. Chandra, L. Tünnemann, T. Löfstedt, and R. Gratz. Transformer-based deep learning for predicting protein properties in the life sciences. *Elife*, 2023.
- H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman. MaskGIT: Masked generative image transformer. In *Conference on Computer Vision and Pattern Recognition*, 2022.
- H. Chang, H. Zhang, J. Barber, A. Maschinot, J. Lezama, L. Jiang, M.H. Yang, K. P. Murphy, W. T. Freeman, M. Rubinstein, Y. Li, and D. Krishnan. Muse: Text-to-image generation via masked generative transformers. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.
- L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *arXiv*, 2021.
- M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, 2020.
- Gene Ontology Consortium. "the gene ontology (GO) project in 2006". *Nucleic Acids Residues*, 2006.
- J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Defossez. Simple and controllable music generation. In *Advances in Neural Information Processing Systems*, 2023.
- J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. Ragotte, L. Milles, B. Wicky, A. Courbet, R. Haas, N. Bethel, P. Leung, T. Huddy, S. Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. Bera, N.P. King, and D. Baker. Robust deep learning-based protein sequence design using proteinmpnn. *Science*, 2022.
- R. R. "Eguchi, C. Choe, and P.-S. Huang. "Ig-VAE: Generative modeling of protein structure by direct 3D coordinate generation". *PLoS Computational Biology*", 2022.
- O.-E. Ganea, X. Huang, C. Bunne, Y. Bian, R. Barzilay, T. S. Jaakkola, and A. Krause. Independent SE(3)-equivariant models for end-to-end rigid protein docking. In *International Conference on Learning Representations*, 2022.
- Z. Gao, C. Tan, and S. Z. Li. VQPL: Vector quantized protein language. *arXiv*, 2023.
- Team Gemini, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- V. Gligorijević, P. D. Renfrew, T. Kosciolk, J. K. Leman, D. Berenberg, T. Vatanen, C. Chandler, B. C. Taylor, I. M. Fisk, H. Vlamakis, R. J. Xavier, R. Knight, K. Cho, and R. Bonneau. Structure-based protein function prediction using graph convolutional networks. *Nature Communications*, 2021.
- A. Herbert and M. Sternberg. MaxCluster: a tool for protein structure comparison and clustering. *arXiv*, 2008.
- J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, O. Vinyals, J. Rae, and L. Sifre. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, 2022.
- A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.
- C. Hsu, R. Verkuil, J. Liu, Z. Lin, B. Hie, T. Sercu, A. Lerer, and A. Rives. Learning inverse folding from millions of predicted structures. In *International Conference on Machine Learning*, 2022.
- J. Ingraham, V. Garg, R. Barzilay, and T. Jaakkola. Generative models for graph-based protein design. In *Advances in Neural Information Processing Systems*, 2019.

- J. Jänes and P. Beltrao. Deep learning for protein structure prediction and design—progress and applications. *Molecular Systems Biology*, 2024.
- B. Jing, S. Eismann, P. Suriana, R. J. L. Townshend, and R. Dror. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*, 2021.
- J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 2021.
- H. Khakzad, I. Igashov, A. Schneuing, C. Goverde, M. Bronstein, and B. Correia. A new age in protein design empowered by deep learning. *Cell Systems*, 2023.
- D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.
- L. F. Krapp, L. A. Abriata, F. Cortés Rodriguez, and M. Dal Peraro. Pesto: parameter-free geometric deep learning for accurate prediction of protein binding interfaces. *Nature Communications*, 2023.
- T. Kucera, C. Oliver, D. Chen, and K. Borgwardt. Proteinshake: Building datasets and benchmarks for deep learning on protein structures. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.
- Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, A. dos Santos Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022.
- H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *Advances in neural information processing systems*, 2024.
- I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.
- F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations*, 2024.
- C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton. CATH – a hierarchic classification of protein domain structures. *Structure*, 1997.
- A. Radford and K. Narasimhan. Improving language understanding by generative pre-training. *arXiv*, 2018.
- A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Better language models and their implications. *Technical report, OpenAI*, 2019.
- A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 2021.
- A. Razavi, A. van den Oord, and O. Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *Advances in Neural Information Processing Systems*, 2019.
- G. Richard, B. P. de Almeida, H. Dalla-Torre, C. Blum, L. Hexemer, P. Pandey, S. Laurent, M. Lopez, A. Laterre, M. Lang, et al. Chatnt: A multimodal conversational agent for dna, rna and protein tasks. *bioRxiv*, 2024.
- L. Robinson, T. Atkinson, L. Copoiu, P. Bordes, T. Pierrot, and T. Barrett. Contrasting sequence with structure: Pre-training graph representations with PLMs. In *NeurIPS 2023 AI for Science Workshop*, 2023.

- N. Sapoval, A. Aghazadeh, M. G. Nute, D. A. Antunes, A. Balaji, R. Baraniuk, C. J. Barberan, R. Dannenfelser, C. Dun, M. Edrisi, et al. Current progress and open challenges for applying deep learning across the biosciences. *Nature Communications*, 2022.
- D. E. Scott, A. R. Bayly, C. Abell, and J. Skidmore. Small molecules, big targets: drug discovery faces the protein–protein interaction challenge. *Nature Reviews Drug Discovery*, 2016.
- J. Su, C. Han, Y. Zhou, J. Shan, X. Zhou, and F. Yuan. Saprot: Protein language modeling with structure-aware vocabulary. In *International Conference on Learning Representations*, 2024.
- B. Trippe, J. Yim, D. Tischer, D. Baker, T. Broderick, R. Barzilay, and T. S. Jaakkola. Diffusion probabilistic modeling of protein backbones in 3D for the motif-scaffolding problem. In *International Conference on Learning Representations*, 2023.
- T. Tu, S. Azizi, D. Driess, M. Schaeckermann, M. Amin, P. Chang, A. Carroll, C. Lau, R. Tanno, I. Ktena, and o. others. Towards generalist biomedical AI. *NEJM AI*, 2024.
- A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.
- M. van Kempen, S. S. Kim, C. Tumescheit, M. Mirdita, J. Lee, C. L. M. Gilchrist, and J. Söding. Fast and accurate protein structure search with foldseek. *Nature Biotechnology*, 2024.
- A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.
- J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, N. Hanikel, S. J. Pellock, A. Courbet, W. Sheffler, J. Wang, P. Venkatesh, I. Sappington, S. Vázquez Torres, A Lauko, V. De Bortoli, E. Mathieu, S. Ovchinnikov, R. Barzilay, T. S. Jaakkola, F. DiMaio, M. Baek, and D. Baker. De novo design of protein structure and function with RFDiffusion. *Nature*, 2023.
- K. E. Wu, K. Yang, R. van den Berg, S. Alamdari, J. Y. Zou, A. X. Lu, and A. P. Amini. Protein structure generation via folding diffusion. *Nature Communications*, 2024.
- J. Xu and Zhang Y. How significant is a protein structure similarity with TM-score = 0.5? *Bioinformatics*, 2010.
- Zhang Y. and Skolnick J. Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic Acids Residues*, 2005.
- L. Yang, Y. Tian, M. Xu, Z. Liu, S. Hong, W. Qu, W. Zhang, B. Cui, M. Zhang, and J. Leskovec. VQGraph: Rethinking graph representation space for bridging GNNs and MLPs. In *International Conference on Learning Representations*, 2024a.
- L. Yang, S. Xu, A. Sellergren, T. Kohlberger, Y. Zhou, I. Ktena, A. Kiraly, F. Ahmed, F. Hormozdiari, T. Jaroensri, E. Wang, E. Wulczyn, et al. Advancing multimodal medical capabilities of gemini. *arXiv*, 2024b.
- J. Yim, B. Trippe, V. De Bortoli, E. Mathieu, A. Doucet, R. Barzilay, and T. Jaakkola. SE(3) diffusion model with application to protein backbone generation. In *International Conference on Machine Learning*, 2023.
- J. Yu, X. Li, J. Y. Koh, H. Zhang, R. Pang, J. Qin, A. Ku, Y. Xu, J. Baldridge, and Y. Wu. Vector-quantized image modeling with improved VQGAN. In *International Conference on Learning Representations*, 2022a.
- J. Yu, Y. Xu, J. Y. Koh, T. Luong, G. Baid, Z. Wang, V. Vasudevan, A. Ku, Y. Yang, B. K. Ayan, B Hutchinson, W. Han, Z. Parekh, X. Li, H. Zhang, J. Baldridge, and Y. Wu. Scaling autoregressive models for content-rich text-to-image generation. *Transactions on Machine Learning Research*, 2022b.
- A. Ziv, I. Gat, G. Le Lan, T. Remez, F. Kreuk, J. Copet, A. Défossez, G. Synnaeve, and Y. Adi. Masked audio generation using a single non-autoregressive transformer. In *International Conference on Learning Representations*, 2024.

A Appendix

A.1 Related Works

Learning from Protein Structures. Learning from protein structure is thriving research field that encompasses a wide variety of tasks crucial task. For instance, inverse folding [Ingraham et al., 2019, Hsu et al., 2022, Dauparas et al., 2022, Jing et al., 2021] or binding site estimation [Krapp et al., 2023, Ganea et al., 2022] are critically enabling for drug design [Scott et al., 2016]. Others [Consortium, 2006, Robinson et al., 2023, Kucera et al., 2023, Gligorijević et al., 2021] focus on learning from protein structure to provide a better understanding of the role of the structure, hence pushing the knowledge frontier of biological processes.

Representation Learning of Protein Structures. [Eguchi et al., 2022] learns a VAE using as inputs matrix distances and predicting 3D coordinates. The supervision is done by comparing distance matrices. Despite the straightforward nature of sampling from a VAE latent space, it often yields subpar sample quality, see [Kingma et al., 2016]. Foldseek [van Kempen et al., 2024] propose to learn a VQ-VAE on small chunks of protein comparing true and reconstructed local geometrical features. Because of the locality of the data used in Foldseek, decoding only small chunks of structures, this quantization defines an interesting tool for efficient structure database search but does not enable full backbone structure sampling. Closer to our work, Gao et al. [2023] propose a quantized autoencoder of protein structures, with however only limited reconstruction performances limiting its applicability.

Generation of Protein Structures. A substantial body of literature addresses the challenge of sampling the protein structure space. Numerous studies have advocated for the use of diffusion-based models Wu et al. [2024], Yim et al. [2023], Watson et al. [2023] or flow matching techniques [Bose et al., 2024]. While many of these works, such as those by [Yim et al., 2023, Watson et al., 2023, Bose et al., 2024], employ complex architectures to ensure invariance to rigid-body transformations, [Wu et al., 2024] opted for an alternative parameterization directly preserving symmetries allowing the authors to capitalize on conventional architectures, yet working only on small protein crops.

A.2 Architectures

We provide in this section additional details on the autoencoder architecture.

Quantizer For the FSQ quantizer, we use linear projections for encoding and decoding of the codes following the original work of [Mentzer et al., 2024]. For all the experiments, we fix the dimension of the codes to $d = 6$. Then, we quantize each channel into L unique values L_1, \dots, L_d and refer to the quantization levels as $\mathbf{L} = [L_1, \dots, L_d]$. The size of the codebook \mathcal{C} is given by the product of the quantization levels: $|\mathcal{C}| = \prod_{j=1}^d L_j$. For the experiments with small codebooks with $|\mathcal{C}| = 4096$, we use $\mathbf{L} = [4, 4, 4, 4, 4, 4]$ and for large codebooks experiments, $|\mathcal{C}| = 64000$, we take $\mathbf{L} = [8, 8, 8, 5, 5, 5]$.

In more details the FSQ quantizer writes as Algorithm 1:

Algorithm 1 Finite Scalar Quantization

- 1: **Input:** $\mathbf{z}_i \in \mathbb{R}^c$ (residue embedding), $W_{proj} \in \mathbb{R}^{c \times d}$ $W_{up-proj} \in \mathbb{R}^{d \times c}$ (weight matrices), \mathbf{L} number of levels
 - 2: **Output:** $\tilde{\mathbf{z}}_i$ (Quantized output)
// Compute low dimensional embedding
 - 3: $\mathbf{z}_i = W_{proj} \cdot \mathbf{z}_i$
// Bound each element z_{ij} between $[-L_j/2, L_j/2]$
 - 4: $z_{ij} = \frac{L_j}{2} \tanh(z_{ij})$
// Round each element to the nearest integer
 - 5: $\tilde{\mathbf{z}}_i = W_{up-proj} \cdot \text{round}(\mathbf{z}_i)$
 - 6: **return** $\tilde{\mathbf{z}}_i$
-

The product $W_{proj} \mathbf{z}_i$ facilitates scalar quantization within a lower-dimensional latent space, thereby defining a compact codebook. This is similar to the low dimensional space used for code index

lookup in Yu et al. [2022a]. The subsequent up-projection operation then restores the quantized code to its original dimensionality.

Resampling The cross-attention based resampling layer can be used for both down (resp. up) sampling, effectively reduces (resp. increases) the length of the sequence of embeddings is described in Algorithm 2.

Algorithm 2 Resampling Layer

- 1: **Input:** Queries, Inputs, Mask
 - 2: **Output:** Queries, Inputs
 - 3: $Queries = \text{CrossAttention}(Inputs, Queries, Mask)$
 - 4: $Queries = \text{MLP}(Queries)$ and $Inputs = \text{MLP}(Inputs)$
 - 5: **return** (Queries, Inputs)
-

Local Cross-Attention Masking The encoder network used in this work preserves the notion of residue order, as defined in the primary structure of a protein (i.e. its ordered sequence of amino acids). We do not provide our algorithm with information regarding the amino-acids. However, we do include the order of the residues from which extract the atoms coordinates. When downsampling the encoder representations using a standard cross-attention operation, the resulting output virtually includes information from any other residue embeddings, irrespective of their relative position in the sequence. In order to encourage the downsampled representation to carry local information, we propose to use local masks in the CrossAttention update of the resampling layer defined in Algorithm 2. This will guide the network towards local positions (in the sequence) and prevent information from distant embeddings. We illustrated the local masking in Figure 4, where only the direct neighbors are kept in the attention update.

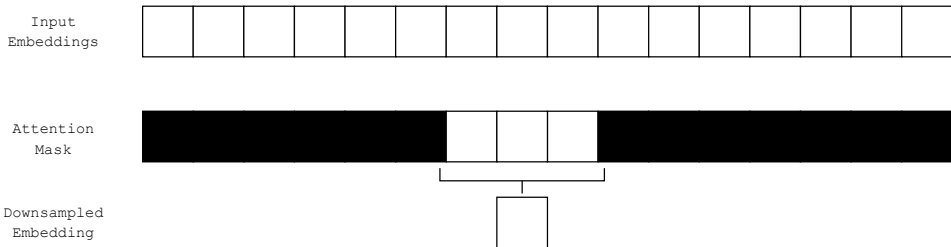


Figure 4: Illustration of the local attention mechanism when using 2 neighbors for aggregation.

Decoder For the decoder, we re-purpose the Structure Module (SM) of AlphaFold [Jumper et al., 2021]. In Jumper et al. [2021], a pair of 1D and 2D features (called the *single representation* and *pair representation* respectively) is extracted from the data by the Evoformer and fed to the SM to reconstruct the 3D structure. Contrary to AlphaFold, we encode the structures with a set of 1D features - the sequence of discrete codes obtained after tokenization. Inspired by the OuterProductMean module of the Evoformer (Alg. 10 in SM of Jumper et al. [2021]), we compute a pairwise representation of the structure by computing the outer product of the quantized sequence after projection, and concatenating the mean with the pair relative positional encoding, as defined in Algorithm 3.

Algorithm 3 Pairwise Module

- 1: **Input:** $s = (s_i)_{i \leq N}$
 - 2: **Output:** $k = (k_{ij})_{i,j \leq N}$
// linear transforms of the initial embedding
 - 3: $s^{left} = W_{left} \cdot s, \quad s^{right} = W_{right} \cdot s$
// ($n = k$): protein length, d : embedding dim
 - 4: $k = \text{einsum}(nd, kd \rightarrow nkd, s^{left}, s^{right})$
 - 5: $k = \text{MLP}(k_{ij}, \text{RelativePositionalEncoding}(i, j)_{i,j \leq N})$
 - 6: **return** $k = (k_{ij})_{i,j \leq N}$
-

Overall algorithm, the encoding and decoding processes write as described in Algorithm 4

Algorithm 4 Overall Algorithm Pseudo-Code

```

1: Input:  $\mathbf{p} \in \mathbb{R}^{n \times 4 \times 3}$ ,  $(\theta, \phi, \psi)$ 
2: Output:  $\tilde{\mathbf{z}}, \tilde{\mathbf{p}}$ 
   // Compute embedding at the residue-level
3:  $\mathbf{z} = \text{GNN}(\mathbf{p})$ 
   // Downsample  $N \rightarrow N/r$ 
4:  $\mathbf{z} = \text{Resampling}(\mathbf{z})$ 
   // Quantize
5:  $\tilde{\mathbf{z}} = q_\phi(\mathbf{z})$ 
   // Upsample  $N/r \rightarrow N$ 
6:  $\mathbf{s} = \text{Resampling}(\tilde{\mathbf{z}})$ 
   // Make pairwise representation for decoding
7:  $\mathbf{k} = \text{PairWiseModule}(\mathbf{s})$ 
   // Decode
8:  $\tilde{\mathbf{p}} = \text{StructureModule}(\mathbf{s}, \mathbf{k})$ 
9: return  $\tilde{\mathbf{z}}, \tilde{\mathbf{p}}$  (Quantized output)

```

A.3 Training Details and Metrics

Dataset We use approximately 310000 entries available in the Protein Data Bank (PDB) [Berman et al., 2000] as training data. The presence of large groups of proteins causes imbalances in the data set, with many protein from the same family sharing structural similarities. To mitigate this issue, we sample the data inversely proportional to the size of the cluster it belongs to when clustering the data by sequence similarity using MMseqs2¹. We filter all chains shorter than 50 residues and crop the structures that have more than 512 residues by randomly selecting 512 consecutive amino acids in the corresponding sequences and resort to cluster size based sampling, see Appendix A.3. We randomly select 90% of clusters training and use the remaining as test set. Amongst these 10% withhold clusters, we retain 20% for validation, the remaining 80% structural clusters being used for test.

Data Preprocessing We consider the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of a set of vertices - or nodes - \mathcal{V} (the residues) with features $f_V^1 \dots f_V^{|V|}$ and a set of edges \mathcal{E} with features $f_E^1 \dots f_E^{|E|}$. For the node features, we use a sinusoidal encoding of sequence position such that for the i -th residue, the positional encoding is $(\phi(i, 1) \dots \phi(i, d))$ where d is the embedding size. For the edge features we follow Ganea et al. [2022]. More specifically, for each defined by residue v_i , a local coordinate system is formed by (a) the unit vector t_i pointing from the α -carbon atom to the nitrogen atom, (b) the unit vector u_i pointing from the α -carbon to the carbon atom of the carboxyl ($-\text{CO}-$) and (c) the normal of the plane defined by t_i and u_i : $n_i = \frac{u_i \times t_i}{\|u_i \times t_i\|}$. Finally, setting: $q_i = n_i \times u_i$, the edge features are then defined as the concatenation of the following:

- relative positional edge features: $p_{j \rightarrow i} = (n_i^T u_i^T q_i^T)(x_j - x_i)$,
- relative orientation edge features: $q_{j \rightarrow i} = (n_i^T u_i^T q_i^T)n_j$, $k_{j \rightarrow i} = (n_i^T u_i^T q_i^T)u_j$, $t_{j \rightarrow i} = (n_i^T u_i^T q_i^T)v_j$,
- distance-based edge features, defined as radial basis functions: $f_{j \rightarrow i, r} = e^{-\frac{\|x_j - x_i\|^2}{2\sigma_r^2}}$, $r = 1, 2 \dots R$ where $R = 15$ and $\sigma_r = 1.5$.

Compression Factor We define the compression factor of Table 1 as the ratio between the number of bits necessary for encoding the backbone position atoms and the number of bits necessary to store the latent codes. With positions stored as 64-bit floats and latent codes stored as 16-bit unsigned

¹The cluster size is readily available in the PDB data set: <https://www.rcsb.org/docs/grouping-structures/sequence-based-clustering>

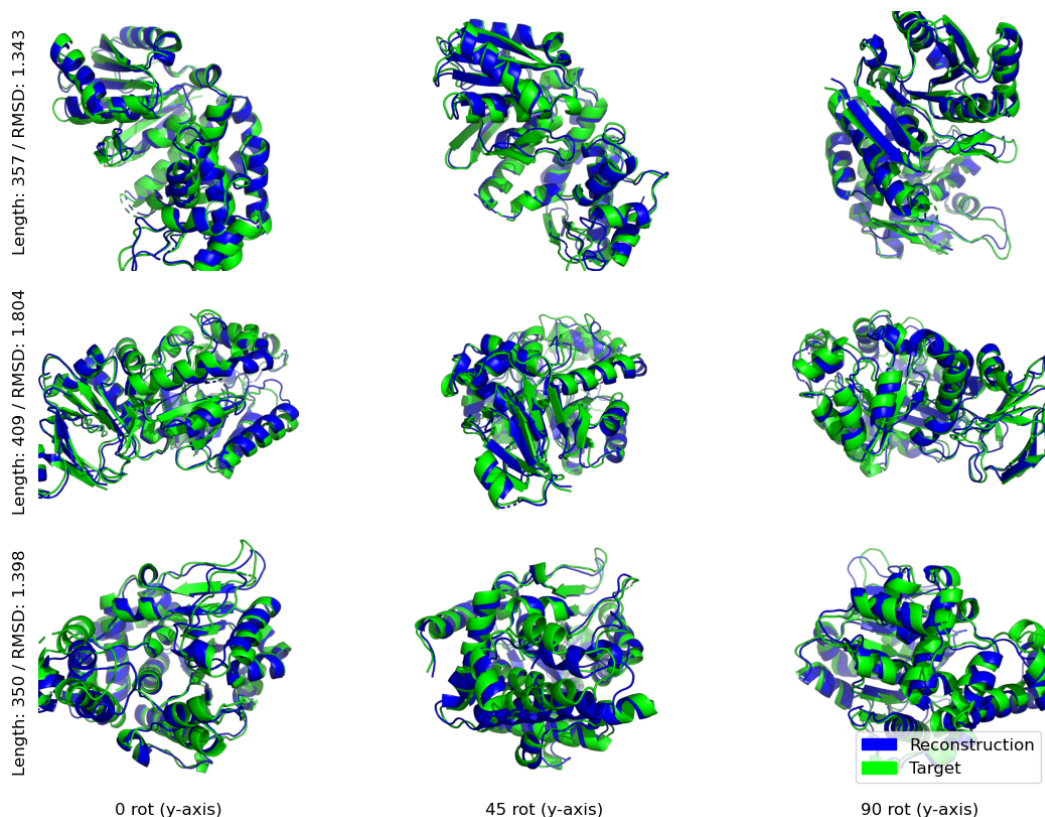


Figure 5: Visualisation of the model reconstruction (blue) super-imposed with the original structures (green) for a downsampling factor of $r = 2$ and $K = 64000$ codes (fourth column of Table 1). Each row shows a different structures seen from a different rotation angle (column). The length and reconstruction RMSD are also given on the left of the most left column.

integers (since the highest number of latent codes is $64000 < 2^{16}$), we can write the compression factor as:

$$\text{Compression Factor} = \frac{N \times 4 \times 3 \times 64}{N \times \log_2(\# \text{Codes})/r} \quad (5)$$

Model Hyperparameters Our encoder is a 3 layers message passing neural network following Dauparas et al. [2022] and with `swish` activation function. The graph sparsity is set to 50 neighbours per residue. When the downsampling ratio is $r > 1$, the resampling operation consists in a stack of 3 resampling layers as described in Algorithm 2, the initial queries being defined as the positional encodings. We strictly follow the implementation of AlphaFold [Jumper et al., 2021] regarding the structure module and use 6 structure layers.

Optimization and Training Details We found that clamping the FAPE loss with a threshold of 10 improves the training stability. The optimization is conducted using AdamW [Loshchilov and Hutter, 2019] with $\beta_1 = 0.9$, $\beta_2 = 0.95$ and a weight decay of 0.1. We use a learning rate warm-up scheduler, progressively increasing the learning rate from 10^{-6} to 10^{-3} , and train the model for 250 epochs on 128 TPU v4-8 with a batch size 1024.

Additional Results We provide in Figure 5 the superimposition of reconstructed structures with their original target.

A.4 De novo Structure Generation

A.4.1 Training details

GPT hyperparameters We use a standard decoder only transformer following the implementation of [Hoffmann et al., 2022] with pre-layer normalization and a dropout rate of 10% during training. We follow Hoffmann et al. [2022] for the parameters choice with 20 layers, 16 heads per layers, a model dimension of 1024 and a query size of 64, resulting in a model with 344M parameters.

Training and Optimization Given the tokenization of PDB training set, we respectively prepend a <bos> and append <eos> tokens and pad all sequences with <pad> token so that all sequences are of size 514. Hence, the maximum number of actual structural token per sequence is 512. The loss associated to <pad> tokens is masked out.

For the optimization, we utilize a ADAMw with $\beta_1 = 0.95$, $\beta_2 = 0.9$ and a weight decay of 0.1. We also employ a learning rate scheduling linearly warming-up increasing the learning rate up to 5.10^{-5} for the first 1000. Following the seminal work of Radford and Narasimhan [2018], we employ embedding, residual and attention dropout with rate of 10% rate. We found the batch size to have a crucial importance on the optimization and adopt a batch size 65,792 tokens per batch.

The total number of actual structural tokens is 70M. In that perspective, and in line with works such as Hoffmann et al. [2022], we believe that leveraging large dataset of predicted structures such as AlphaFold² can provide significant improvements when training the latent generative model.

A.4.2 Structure generation metrics

Designability We adopt the same framework than [Yim et al., 2023, Trippe et al., 2023, Wu et al., 2024] to compute the designability or *self-consistency* score:

- Compute 8 putative sequences from ProteinMPNN [Dauparas et al., 2022] with a temperature sampling of 0.1.
- Fold each of the 8 amino-acid sequences using ESMFold [Lin et al., 2022] without recycling, resulting in 8 folds per generated structure.
- Compare the 8 ESMFold-predicted structures with the original sample using either TM-score (scTM) or RMSD (scRMSD). The final score is taken to be the best score amongst the 8 reconstructed structures.

In Table 2, we report the proportion of generated structures that are said to be designable, *i.e* samples for which scTM>0.5 (or scRMSD< 2 Å when using the RMSD).

Novelty For the reference dataset, we use the s40 CATH dataset [Orengo et al., 1997], publicly available here. In order to reduce the computation time, we first retrieve the top $k = 1000$ hits using Foldseek [van Kempen et al., 2024]. We then perform TM-align [Y. and J., 2005] for each match against the targeted sample and report the TM-score corresponding to the best hit. A structure is then considered as novel if the maximum TM-score against CATH (cathTM) is lower than 0.5 and report the proportion of novel structures in Table 2

Diversity Finally, we measure the diversity of the samples similarly to Watson et al. [2023]. More specifically, the generated samples are clustered using an all-to-all pairwise TM-score as the clustering criterion and we observe the resulting number of structural clusters normalized by the number of generated samples. For a diverse set of generated samples, each cluster should be composed of only a few samples - or equivalently, the number of different clusters should be high. We use MaxCluster Herbert and Sternberg [2008] with a TM-score threshold of 0.6 as in Watson et al. [2023].

A.4.3 Sampling

Baselines For each baseline methods, we follow a standardized process similar to that of Yim et al. [2023] to generate the testing dataset: we sample 8 backbones for every length between 100 and

²<https://alphafold.ebi.ac.uk/>

500 with length step of 5: [100, 105, . . . , 500]. We re-use the publicly available codes and use the parameters reported in Watson et al. [2023] and Yim et al. [2023] respectively.

Generating Structures with a Decoder-Only Transformer Sampling for our method is a 2 steps process: first, sample a sequence of structural tokens from the trained prior described in Appendix A.4.1, then reconstruct the structures using the trained decoder. There are many ways to sample from a decoder-only transformer model [Vaswani et al., 2017, Radford and Narasimhan, 2018, Radford et al., 2019, Holtzman et al., 2020]. We chose temperature sampling [Vaswani et al., 2017] with other alternative sampling strategies such as top-k [Radford et al., 2019] and top-p (or nucleus) sampling [Holtzman et al., 2020] resulting in little improvement at the cost of increased complexity. As showed in Vaswani et al. [2017], the temperature controls the trade-off between the confidence and the diversity of the samples. In order to tune the temperature, we sampled 2000 samples for each temperature between 0.2 and 0.8 in step of 0.2 and compute the designability score for these samples. As expected, the higher the temperature, the more varied the samples. Indeed, the distribution of the proteins length, depicted in Figure 6, shows a greater diversity of higher temperatures. On the other hand, with lower temperatures, the length distribution is more concentrated around few lengths. Similarly we can see than for lower temperatures, the samples closer to the modes of the length distribution (samples with length between 100 and approximately 300) achieve higher scTM-score (see Figure 6). The results reported in Table 2 are obtained with a temperature of 0.6, as it achieves a satisfying trade-off between designability and diversity.

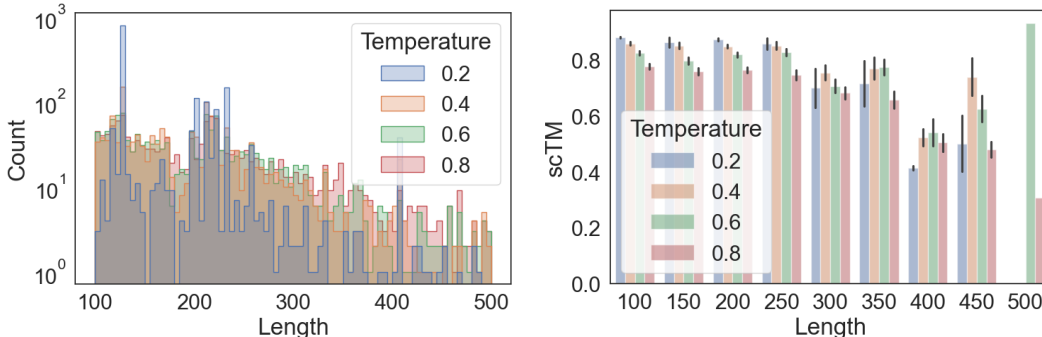


Figure 6: Ablation of the temperature sampling. Left: Histogram of the generated structure length. Right: Designability score vs temperature sampling.

Contrary to the baselines, our model learns the joint distribution the length and the structures $p(s) = \int_l p(s, l) dl$ where the random variables s and l represent the structures and the length respectively. Indeed, only the conditional distribution $p(s|l)$ is modeled by the diffusion-based baselines. In Figure 6, we show the length distribution learned by the model. Since we can only sample from the joint distribution and not the conditional, we adopt the following approach: First, we sample 40,000 structures from the model (using a temperature of 0.6 as previously established). We then bin the generated structures by length, with a bin width of 5 and bin centers uniformly distributed between 100 and 500, specifically: [100, 105, . . . , 500]. Finally, we limit the maximum number of structures per bin to 10, randomly selecting 10 structures if a bin contains more than this number.

Method	scTM (\uparrow)	scRMSD (\uparrow)	Novelty	Diversity
Ours	76.61%	41.00%	35.78%	71.4 %
FrameDiff	75.77%	25.31%	56.64%	82.0%
RFDiffusion	97.07%	71.14%	86.11%	95.0%
Validation Set	97.67%	82.36 %	—	70.1%

Table 2: Structure generation metrics for our method alongside baselines (and nature) specifically designed for protein structure generation. Self-consistent TM-score (scTM) and self-consistent RMSD (scRMSD) are two different ways to assess the designability of the generated structure. Note that while high novelty score is desirable, structures that are too far from the reference dataset can also be a sign of unfeasible proteins.

A.4.4 Additional Results

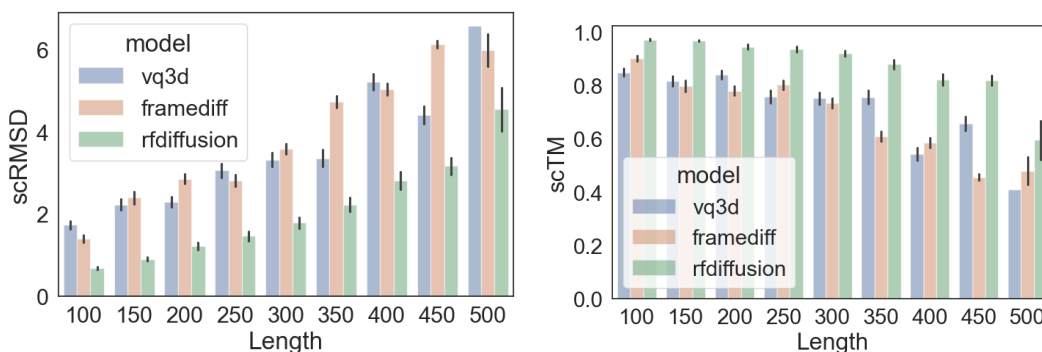


Figure 7: Designability score vs samples length. Left: scRMSD score for different structure lengths. Right: scTM score for different structure lengths.

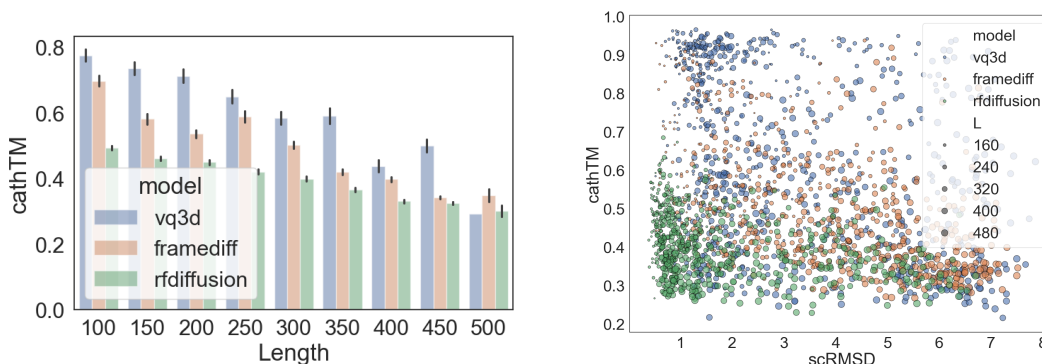


Figure 8: Left: Novelty score for different structure lengths. Right: Novelty score *versus* designability score.

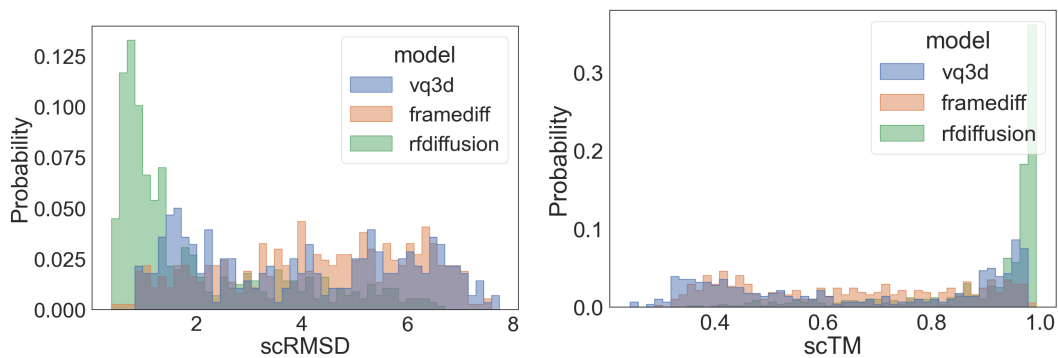


Figure 9: Distribution of the designability scores for novel domains (cathTM<0.5).