

InstructPLM: Aligning Protein Language Models to Follow Protein Structure Instructions

Jiezhong Qiu

jiezhongqiu@outlook.com

Zhejiang Lab and Zhejiang University

Junde Xu

Zhejiang Lab and The Chinese University of Hong Kong

Jie Hu

Zhejiang Lab

Hanqun Cao

The Chinese University of Hong Kong

Liya Hou

Zhejiang Lab

Zijun Gao

Zhejiang Lab

Xinyi Zhou

The Chinese University of Hong Kong

Anni Li

Nanjing Normal University

Xiujuan Li

Nanjing Normal University

Bin Cui

Zhejiang Lab

Fei Yang

Zhejiang Lab <https://orcid.org/0000-0003-4802-3191>

Shuang Peng

Zhejiang Lab

Ning Sun

Zhejiang Lab

Fangyu Wang

Zhejiang Lab

Aimin Pan

Zhejiang Lab

Jie Tang

Tsinghua University

Jieping Ye

Alibaba

Junyang Lin

Alibaba

Jin Tang

Zhejiang Lab

Xingxu Huang

Zhejiang Lab

Pheng Ann Heng

Chinese University of Hong Kong

Guangyong Chen

Zhejiang Laboratory <https://orcid.org/0000-0002-5892-8608>

Biological Sciences - Article

Keywords: Protein Design, Protein Language Models, Cross-Modal Alignment, Instruction Fine-Tuning

Posted Date: July 1st, 2024

DOI: <https://doi.org/10.21203/rs.3.rs-4432242/v1>

License:  This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: There is **NO** Competing Interest.

InstructPLM: Aligning Protein Language Models to Follow Protein Structure Instructions

Jiezhong Qiu^{1,2*†}, Junde Xu^{1,4†}, Jie Hu^{1†}, Hanqun Cao^{4†},
Liya Hou¹, Zijun Gao¹, Xinyi Zhou⁴, Anni Li³, Xiujuan Li³,
Bin Cui¹, Fei Yang¹, Shuang Peng¹, Ning Sun¹, Fangyu Wang¹,
Aimin Pan¹, Jie Tang⁵, Jieping Ye⁶, Junyang Lin⁶, Jin Tang¹,
Xingxu Huang¹, Pheng Ann Heng^{1,4}, Guangyong Chen^{1*}

¹*Zhejiang Lab.

²Zhejiang University.

³Nanjing Normal University.

⁴The Chinese University of Hong Kong.

⁵Tsinghua University.

⁶Alibaba.

*Corresponding author(s). E-mail(s): jiezhongqiu@zhejianglab.com;
gychen@zhejianglab.com;

[†]These authors contributed equally to this work.

Abstract

Large language models are renowned for their efficacy in capturing intricate patterns, including co-evolutionary relationships, and underlying protein languages. However, current methodologies often fall short in illustrating the emergence of genomic insertions, duplications, and insertion/deletions (indels), which account for approximately 14% of human pathogenic mutations. Given that structure dictates function, mutated proteins with similar structures are more likely to persist throughout biological evolution. Motivated by this, we leverage cross-modality alignment and instruction fine-tuning techniques inspired by large language models to align a generative protein language model with protein structure instructions. Specifically, we present a method for generating variable-length and diverse proteins to explore and simulate the complex evolution of life, thereby expanding the repertoire of options for protein engineering. Our proposed protein LM-based approach, InstructPLM, demonstrates significant performance enhancements both in silico and in vitro. On native protein backbones, it achieves

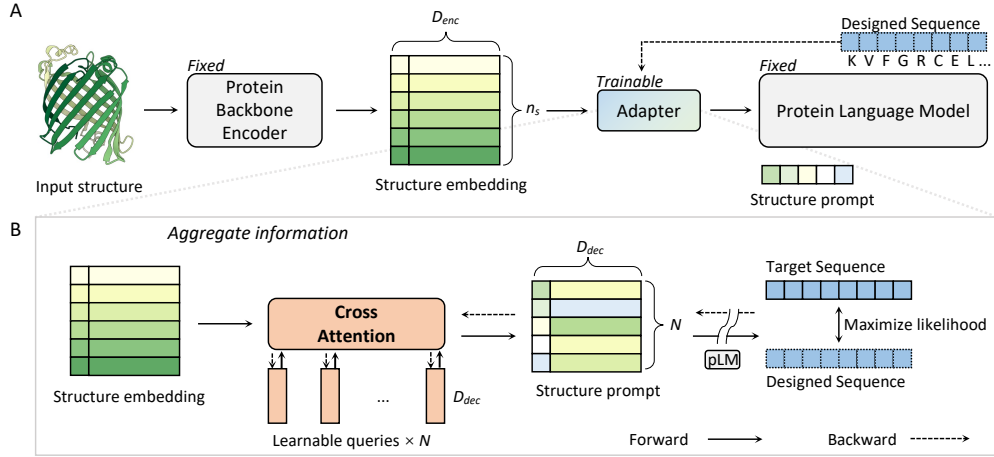


Fig. 1 (A) Illustration of proposed method, InstructPLM. The network consists of three parts, the protein backbone encoder, the protein structure-sequence adapter, and the protein sequence decoder. The protein structure-sequence adapter is the only trainable part while the parameters of the structure encoder and sequence decoder are fixed during training. (B) Detailed illustration of the structure-sequence adapter. The adapter adopts a structure embedding as input, incorporating the structure information with a set of learnable queries. The queries will form as the structure prompt guiding the generation of the protein sequence decoder.

a perplexity of 2.68 and a sequence recovery rate of 57.51, surpassing Protein-MPNN by 39.2% and 25.1%, respectively. Furthermore, we validate the efficacy of our model by redesigning PETase and L-MDH. For PETase, all fifteen designed variable-length PETase exhibit depolymerization activity, with eleven surpassing the activity levels of the wild type. Regarding L-MDH, an enzyme lacking an experimentally determined structure, InstructPLM is able to design functional enzymes with an AF2-predicted structure. Code and model weights of InstructPLM are publicly available*.

Keywords: Protein Design, Protein Language Models, Cross-Modal Alignment, Instruction Fine-Tuning

1 Introduction

Mutations, encompassing point mutations arising from single nucleotide changes, as well as deletions, duplications, and insertions of multiple bases, serve as the driving force of evolution, facilitating the rapid adaptation of life to continuously changing environmental conditions as the fundamental cause of genetic diversity. These genetic variations provide a rich repository of proteins for reference and selection

*Model weights: <https://huggingface.co/InstructPLM/MPNN-ProGen2-xlarge-CATH42>.
Code: <https://github.com/Eikor/InstructPLM>.

in protein engineering. A quintessential challenge in protein engineering [1] is protein sequence design, also known as protein inverse folding, which entails identifying amino acid sequences capable of folding into a specified protein backbone structure. Highly accurate protein sequence design enables the generation of more effective enzymes [2, 3], improved protein-based therapeutics [4], and engineered proteins for industrial purposes like biofuels production and environmental remediation [5, 6]. The recent emergence of deep learning-based methods, exemplified by ProteinMPNN [7] and ESM-IF [8], has significantly advanced the field of protein sequence design. These methods formalize protein sequence design as a *multimodal learning* problem and solve the problem by training models with an encoder-decoder pattern which translates protein backbone structure into corresponding sequences. The swift evolution of deep learning architectures, including graph neural networks [7], and Transformers [9], coupled with the availability of high-quality protein structure data from Protein Data Bank (PDB) and AlphaFoldDB, has catalyzed substantial advancements in this field. For example, myoglobin and tobacco etch virus (TEV) protease designed by ProteinMPNN have demonstrated superior expression, stability, and functionality [10]; The de novo binders designed by ProteinMPNN exhibit high affinity and specificity to targets [11].

Multimodal learning, while straightforward and effective, faces a significant hurdle due to the paucity of high-quality cross-modal data. For instance, in the domain of visual-language understanding, training models from scratch necessitates access to well-curated datasets of image-text pairs, such as those of image captioning and visual question answering. Similarly, in protein sequence design, the training of the aforementioned ProteinMPNN depends on protein assemblies experimentally determined by X-ray crystallography or cryoEM in Protein Data Bank (PDB). To tackle the issue of limited data, an emerging strategy in the field of large vision-language models is *cross-modality alignment* [12–16], which is widely speculated to have been integrated into GPT-4 [17], although the technical details of GPT-4 have yet to be publicly disclosed. This technique leverages existing pre-trained unimodal models to facilitate multimodal understanding. Pre-trained unimodal models learn domain-specific representations through extensive training on large unimodal datasets, yet they typically lack the ability to understand and reason across modalities. Cross-modality alignment bridges this gap by carefully aligning a pre-trained visual model with a pre-trained language model, thereby enabling effective cross-modal generation, understanding, and reasoning. Despite the success of cross-modality alignment techniques in vision-language understanding, their potential in protein sequence design remains unexplored due to the inherent dissimilarities between protein science and visual-language tasks. Consequently, addressing this challenge necessitates the development of powerful unimodal models and alignment techniques specifically tailored to proteins.

Fortunately, protein language models (pLMs), such as ESM [18], ProtGPT [19], and ProGen [20, 21], have emerged as a pivotal innovation in bioinformatics and computational biology, particularly in modeling protein sequences. Drawing inspiration from the achievements of large-scale language models like GPT and BERT, pLMs are pre-trained on vast datasets of protein sequences, leading to a comprehensive understanding of the protein universe. This includes the ability to capture the distribution of

93 observed evolutionary sequences, generate novel yet viable sequences, and predict pro-
 94 tein fitness. Specifically, pLMs have demonstrated the capability to generate functional
 95 protein sequences according to certain conditions. For example, GPT-based pLMs
 96 such as ProGen and ProtGPT can generate proteins following homologous samples or
 97 control tags specifying protein properties; ESM-based pLMs [22–24] design desired pro-
 98 tein sequences by applying or sampling from the pre-trained masked language model.
 99 However, unlike general language models which exhibit zero-shot generalization and
 100 the ability to understand user intent on a wide range of tasks through methods like
 101 instruction fine-tuning [25, 26] or reinforcement learning [27, 28], it still remains an
 102 open area of inquiry how pLMs can generate protein sequences following fine-grained
 103 and complex biological instructions and even simulate the evolution of life. As men-
 104 tioned in "Seven technologies to watch in 2024" [29] regarding deep learning for protein
 105 design, "Sequence-based approaches can build on and adapt existing protein features
 106 to form new frameworks, but they're less effective for the bespoke design of structural
 107 elements or features."

108 In this work, we demonstrate the successful adaptation of *cross-modality align-*
 109 *ment* and *instruction fine-tuning* techniques, originally developed for large language
 110 models, to the domain of protein sequence design. This transferability underscores the
 111 broad applicability of these techniques to life science. Our proposed model, Instruct-
 112 PLM, employs a lightweight cross-attention layer to align a frozen protein backbone
 113 encoder with a frozen protein language model decoder, aiming to teach the protein
 114 language model to design sequences following protein structure instructions. This con-
 115 figuration enables us to harness the robust generalization capabilities of pLMs, while
 116 stimulating pLMs to follow residue-level protein structure instructions at the same
 117 time. InstructPLM outperforms existing sequence design techniques in terms of per-
 118 plexity and sequence recovery, while incurring the training of only an additional 2%
 119 of parameters.

120 The overall model architecture of InstructPLM is composed of three primary com-
 121 ponents — a protein language model decoder, a protein backbone encoder, and a
 122 protein structure-sequence adapter, as depicted in Figure 1 and described as follows:

- 123 • **Protein Language Model Decoder:** InstructPLM adopts ProGen2 as its protein
 124 language model. The ProGen2 model family includes auto-regressive language mod-
 125 els pre-trained on around 230 million protein sequences from a mixture of Uniref90
 126 and BFD30 databases, whose model size ranges from 151M to 6.4B. We choose the
 127 largest one, ProGen2-xlarge with 6.4B parameters in InstructPLM. The ablation
 128 study about the size of the protein language model decoder is shown in Section 2.3.1.
- 129 • **Protein Backbone Encoder:** InstructPLM initializes its protein backbone
 130 encoder from other existing protein sequence design models, such as the backbone
 131 encoder of ProteinMPNN and ESM-IF. The ablation study about the choice of
 132 protein backbone encoder is shown in Section 2.3.2.
- 133 • **Protein Structure-Sequence Adapter:** The protein structure-sequence adapter
 134 is a key component of InstructPLM, not only because it is responsible for align-
 135 ing structure and sequence into the same semantic space, but also because it
 136 contains all trainable parameters of InstructPLM. This adapter comprises a single-
 137 layer cross-attention module initialized randomly. The module uses several trainable

embeddings as query vectors and the output of the protein backbone encoder as keys/values for cross-attention computation. This cross-attention module compresses the protein backbone embedding to a fixed-length structure instruction. The ablation about the number of queries (i.e., the length of structure instruction) is shown in Section 2.3.3. Additionally, we follow Qwen-VL [13] to add 1D absolute positional encodings into the cross-attention to keep protein primary structure information during compression. The compressed protein backbone structure feature sequence is subsequently fed into the protein language model as a soft prompt.

Our experimental results show that our InstructPLM framework is effective in protein sequence design in terms of perplexity and sequence recovery on the CATH dataset. Notably, we validate InstructPLM on redesigning PETase and L-MDH. For PETase, all fifteen designed variable-length proteins exhibit depolymerization activity, with twelve surpassing the activity levels of the wild type. For L-MDH, which does not have an experimentally determined structure, we show that InstructPLM is able to design functional enzyme with AlphaFold2 [30]-predicted structure — 3 out of 15 designed L-MDH exhibit detectable enzymatic activity.

2 Results

We firstly evaluate InstructPLM by following the standard in silico protein sequence design benchmark in Section 2.1. The experimental evaluation and ablation studies are then described in Section 2.2 and Section 2.3.

2.1 In Silico Protein Sequence Design

2.1.1 InstructPLM Designs Sequences with High Recovery

For computational evaluation, we assess the performance of InstructPLM using CATH 4.2 [37], which is a widely recognized benchmark in protein sequence design. We adopt the official train, validation, and test splits, which contain 18,024, 608, and 1,120 proteins respectively. Table 1 and the left panel of Figure 2(A) show the detailed sequence design performance on the CATH 4.2 held-out test split.

InstructPLM demonstrates superior performance compared to models trained from scratch on the CATH 4.2 dataset. Regarding perplexity, InstructPLM significantly improves upon the best baseline method, ESM-IF, by 32.3% (from 3.96 to 2.68). In sequence recovery, InstructPLM also shows a significant enhancement, with an 11.3% increase in the recovery rate over the best baseline, PiFold (from 51.66 to 57.51). It’s particularly noteworthy that the comparison between InstructPLM and ProteinMPNN highlights the efficacy of protein language models in representing the distribution of evolutionary sequences. This is evident as they utilize the same backbone encoder, with the primary distinction lying in their sequence decoders.

The recently introduced LM-Design [22] shares a conceptual connection with InstructPLM, as both leverage protein language models (pLMs) for the generation of protein sequences. The distinction between them primarily hinges on their approach to decoding protein sequences. LM-Design employs a modified ESM pLM in a step-wise fashion to iteratively refine a sequence initially decoded by ProteinMPNN or

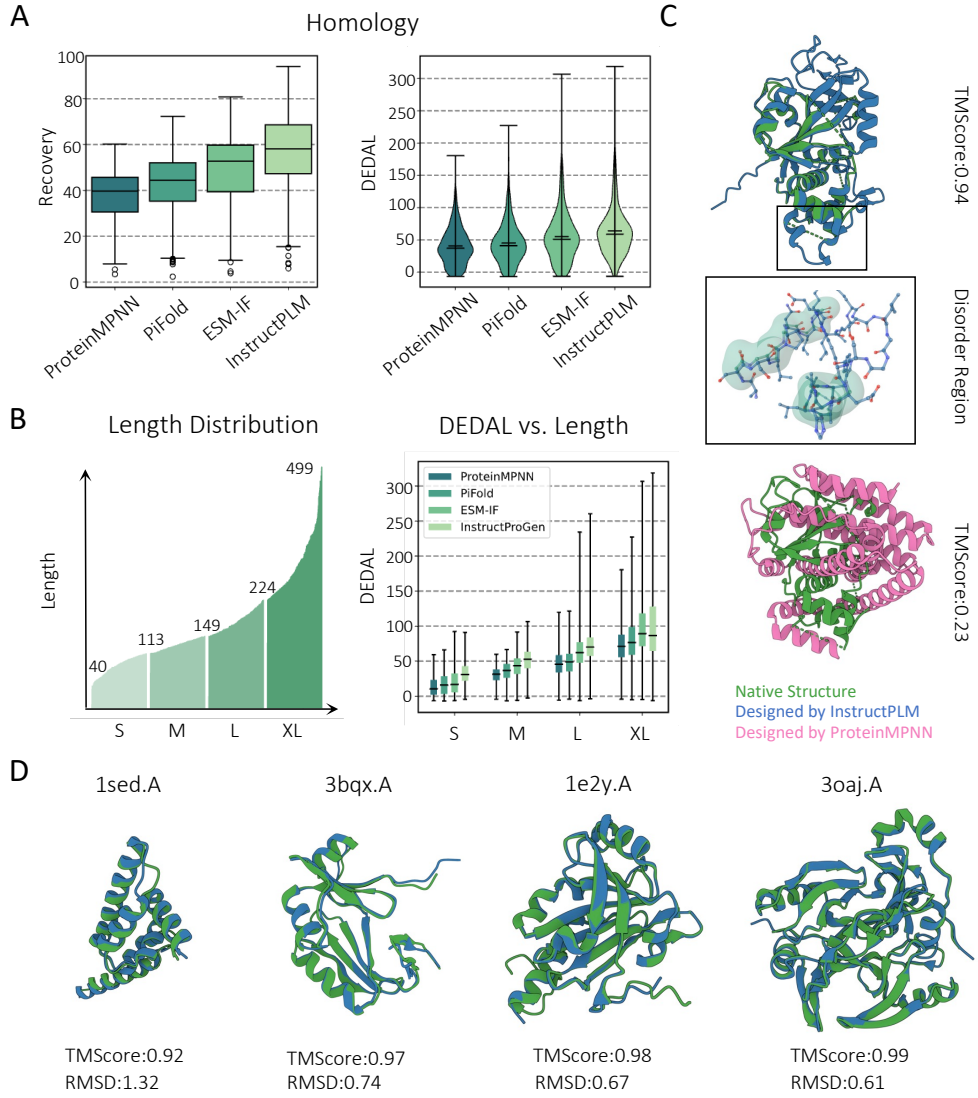


Fig. 2 In silico protein sequence design performance of InstructPLM. **(A)** Recovery rate (left) and homology score predicted by DEDAL (right) on CATH 4.2 test split. Compared to existing methods, InstructPLM achieves the best performance. **(B)** DEDAL score breakdown on different sequence lengths. InstructPLM achieves consistently better performance than other approaches. **(C)** Comparison of predicted structures of the designed sequence of PDB code 4F0D. Compared to ProteinMPNN (TM-Score: 0.23), InstructPLM (TM-Score: 0.94) generates more plausible sequences in the regions where the protein structure is missing. **(D)** Additional designed cases from InstructPLM, using CATH 4.2 test set as the structure templates. All TM-Scores are computed using TM-align [31].

179 PiFold, utilizing the masked language model of ESM. On the other hand, InstructPLM
 180 decodes protein sequences in an autoregressive manner, utilizing the protein backbone

Table 1 Sequence design performance on CATH 4.2 held-out test split. The metrics include perplexity (exponentiated categorical cross-entropy loss per residue) and recovery rate (percentage of correctly predicted residues). The best performance is shown in **bold**, while the best baseline is indicated with an underline. For a fair comparison, we test the performance of ESM-IF on CATH 4.2, although it is originally trained and evaluated on CATH 4.3.

Model	Perplexity on CATH 4.2↓			Recovery Rate on CATH 4.2↑		
	Short	Single-chain	All	Short	Single-chain	All
StructGNN [32]	8.29	8.74	6.40	29.44	28.26	35.91
GraghTrans [32]	8.39	8.83	6.63	28.14	28.46	35.82
GCA [33]	7.09	7.49	6.05	32.62	31.10	37.64
GVP [34]	7.23	7.84	5.36	30.60	28.95	39.47
AlphaDesign [35]	7.32	7.63	6.30	34.16	32.66	41.31
ProteinMPNN [7]	6.21	6.68	4.61	36.35	34.43	45.96
ESM-IF [8]	6.93	6.65	<u>3.96</u>	35.28	33.78	48.95
PiFold [36]	<u>6.04</u>	<u>6.31</u>	4.55	<u>39.84</u>	38.53	51.66
LM-Design-MPNN [22]	7.01	6.58	4.41	35.19	40.00	54.41
LM-Design-PiFold [22]	6.77	6.46	4.52	37.88	<u>42.47</u>	<u>55.65</u>
InstructPLM (Relative Gain)	3.22 (-46.7%)	3.17 (-49.7%)	2.68 (-32.3%)	61.59 (+54.5%)	59.29 (+39.6%)	57.51 (+3.3%)

embedding derived from ProteinMPNN. A comparative analysis of InstructPLM and LM-Design reveals the benefits of the cross-modality alignment and instruction tuning techniques that we have incorporated. These techniques enable a comprehensive exploitation and effective use of structural information in sequence design. Specifically, when comparing InstructPLM with LM-Design-MPNN, which both leverage ProteinMPNN, InstructPLM demonstrates relative improvements of 39.2% (from 4.41 to 2.68) in perplexity and 5.7% (from 54.41 to 57.51) in recovery rate, respectively.

More interestingly, InstructPLM exhibits a consistent pattern of improvement across different subsets of CATH 4.2, and especially achieves more significant enhancements when designing short proteins and single-chain proteins. This observation suggests that the model’s architecture, training techniques, and the pLM’s prior knowledge are particularly effective for these protein types. This also indicates that InstructPLM is not only a powerful general-purpose protein designer but also excels in specialized applications, particularly where the protein sequences are shorter or consist of a single chain.

We also evaluate InstructPLM on TS50 and TS500 datasets, which consist of 50 and 470 proteins and are often employed as additional benchmarks to further test generalization capability [22, 35, 36] beyond CATH dataset. The detailed results are shown in Supplementary Information, where InstructPLM demonstrates consistent and robust performance.

2.1.2 InstructPLM Generates Variable-Length and Homologous Sequences

In contrast to conventional approaches that generate sequences of a fixed length, a distinctive feature of InstructPLM lies in its ability to create protein sequences of variable

lengths through an open-ended generation process. This is facilitated by the aligned pLM decoder. It generates subsequent amino acid tokens in an autoregressive manner, pausing only upon encountering the termination token. This innovative capacity for open-ended generation aligns closely with the intrinsic nature of biological evolution, where a variety of mutations such as insertions, deletions, and their combinations are commonplace. However, the traditional sequence recovery measure is not well-suited for evaluating the quality of variable-length sequences, as it evaluates the per-position accuracy of designed sequences compared to their native counterparts. To effectively assess InstructPLM’s unique capability, we propose to use sequence homology score as a new metric. We employ DEDAL [38], a state-of-the-art deep learning-based method for homology detection. This method has shown a remarkable improvement in alignment accuracy, achieving up to two or three times better performance than traditional methods when searching for remote homologs.

Within the test split of CATH 4.2, we randomly design 20 protein sequences at a temperature setting of 0.15 for each native protein structure. The sequence homology score, as determined by DEDAL, serves as a metric for comparing the similarity between the native and artificially designed sequences. As depicted in the right panel of Figure 2(A), the distribution of homology scores across various methods is showcased, with InstructPLM achieving an average homology score of 63.97, surpassing ProteinMPNN/ESM-IF/PiFold by margins of 23.25/8.83/18.9, respectively. Figure 2(B) provides a detailed breakdown of homology score distributions across sequences of different lengths, demonstrating that InstructPLM maintains a consistently high level of performance in the design of sequences with varying lengths.

2.2 Experimental Evaluation of InstructPLM

Although InstructPLM achieves significant improvement regarding in silico sequence design metrics such as perplexity, sequence recovery, and homology score, we aim to test its ability to design functional proteins. Specifically, we evaluate InstructPLM on PETase (PDB code: 7SH6) and L-MDH (Uniprot: A0A319AA41). Notably, L-MDH has no experimentally determined structure, thus we design L-MDH based on its AF2-predicted structure. The design process is outlined in Figure 3(A). Utilizing the backbone structure as a starting point, InstructPLM produces 10,000 potential sequences via auto-regressive generation with top-p sampling ($p = 0.9$) and a temperature of 0.8, as detailed in Section 4.3. The sequences’ structures are predicted using ESMFold [39] and compared to the native structure with DeepScore from DeepAlign package [40, 41] to calculate TM-Score, which is a measure of structural similarity. Lastly, we choose the top sequences ranked by TM-Score for further experimentation. Upon meticulous experimental evaluation, the sequences designed by our model for the two proteins showed activity, demonstrating the model’s effectiveness. We provide a detailed study of each designed enzyme in Section 2.2.1 and Section 2.2.2

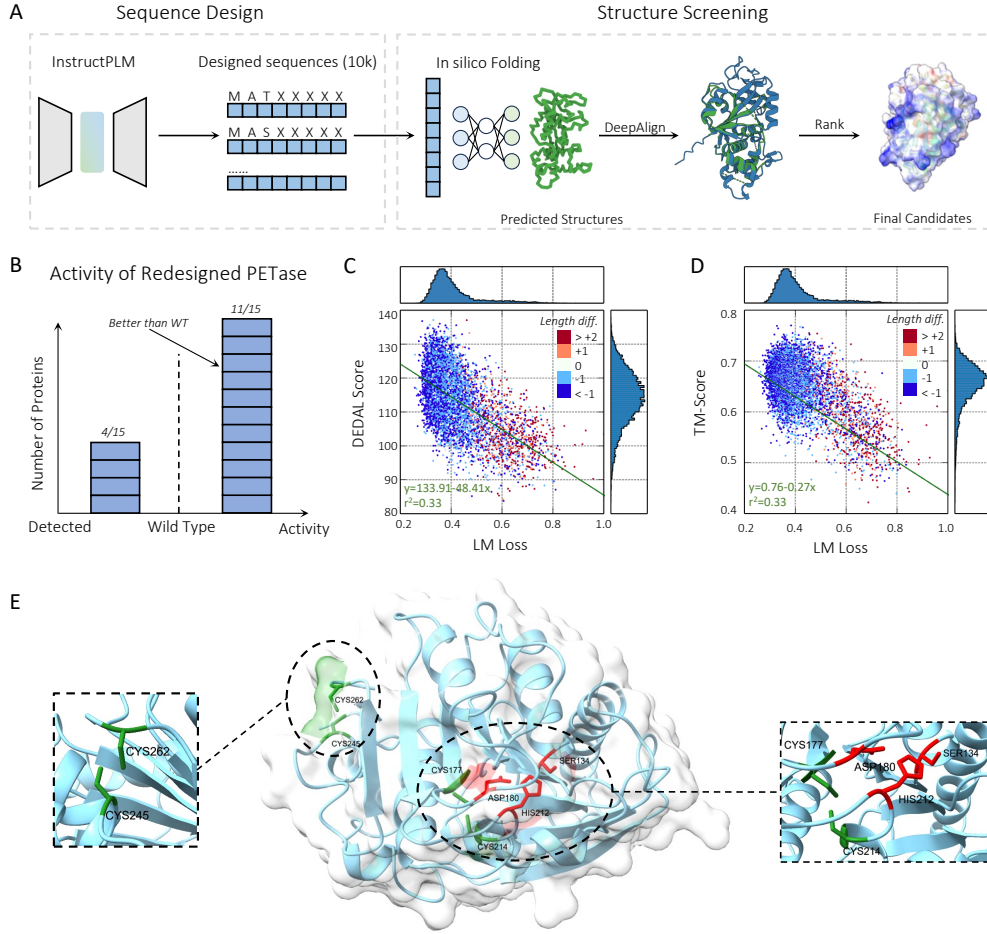


Fig. 3 Experimental evaluation of InstructPLM. **(A)** Overview of the protein design post-processing pipeline using InstructPLM. Given a target structure, InstructPLM samples 10,000 candidate sequences. The structures of all sequences are predicted using ESMFold. DeepAlign is then employed for structure alignment. Finally, samples with higher structural similarity are selected for further experiments. **(B)** Result of PETase. All 15 sequences designed by InstructPLM exhibit high expression levels. **(C)** The joint distribution of the homology score (DEDAL score) and language model loss (LM loss) indicates the overall design quality of the 10,000 candidate sequences. **(D)** The joint distribution of the structure similarity score (TM-Score) and language model loss (LM loss) providing an overview of the 10,000 designed sequences. **(E)** An visualization case of PETase designed by InstructPLM, the protein structure is predicted by ESMFold. It has catalytic triad SER134-HIS212-ASP180 and two disulfide bridges (DS1 and DS2).

2.2.1 PETase

Designing PETase poses a test of InstructPLM’s ability to improve upon naturally occurring proteins. We select the top 15 sequences in terms of TM-Scores for experimentation. The lengths of these 15 sequences vary from 261 to 269, reflecting the InstructPLM’s capability to generate sequences of variable lengths.

When expressed in *Escherichia coli*, all 15 sequences designed by InstructPLM show good expression levels. The PET hydrolysis reaction products — bis (2-hydroxyethyl) terephthalate (BHET), mono (2-hydroxyethyl) terephthalate (MHET), and terephthalic acid (TPA) — are quantified using high-performance liquid chromatography. All 15 sequences designed by InstructPLM have PET hydrolytic activity, while 11 sequences are better than wild-type (PDB code: 5XJH), as shown in Figure 3(B). Sequence alignment, as shown in Supplementary Information, reveals that all 15 designed PETase sequences contain the strictly conserved catalytic triad SER134-HIS212-ASP180. SER134 serves as the nucleophile and is located within hydrogen-bond distance to be polarized by the base HIS212, which is in turn stabilized by acid ASP180 [42]. Additionally, 4 out of 15 sequences have two intramolecular disulfide bridges (DS1:C177-C214 and DS2:C245-C262), and all (15/15) have DS2 bridge connecting the C-terminal helix at the last loop. An example of a sequence generated by InstructPLM featuring both the catalytic triad and the two disulfide bridges is shown in Figure 3(E).

Through this comprehensive analysis, we observe that InstructPLM has effectively learned the significance of catalytic and disulfide bridge residues, which is crucial for enhancing the success rate of designed PETase enzymes. To further our understanding, we perform two regression analyses to explore the correlation between (1) language model loss (LM loss) and homology score (DEDAL score); and (2) LM loss and TM-Score. The results, depicted in Figures 3(C) and 3(D), suggest that LM loss can be a valuable metric for assessing the similarity in both sequence and structure between designed and native proteins.

2.2.2 L-MDH

L-MDH (L-Malate dehydrogenase, Uniprot: A0A319AA41) is a key enzyme in the citric acid cycle with no experimentally determined structure available. We rely on AlphaFold2 [30] predicted structures as the basis for design. Following the same protocol of PETase, we generate 10,000 sequences with InstructPLM, and select 15 sequences with the highest TM-Scores. Out of the 15 designed L-MDH sequences, our experimental validation reveals that three exhibited detectable enzymatic activity. While these active variants do not surpass the performance of the wild-type enzyme, the fact that they demonstrated activity is a significant achievement, given the absence of an experimentally confirmed structure for L-MDH. The ability to generate functional enzymes based solely on computationally predicted structures underscores the potential of combining advanced protein structure prediction tools like AlphaFold [30] and structure design tools such as Chroma [43] with our design methodology. Notably, both three sequences are shorter than wild-type, further demonstrating the flexibility and adaptability of our approach. We compare the L-MDH with three detectable

Table 2 Ablation results on pre-trained pLMs: the perplexity of CATH 4.2, TS50, and TS500 is obtained based on different pre-trained ProGen2 models [45]. The number of trainable parameters and the total parameters are shown. The best performance is shown in **bold**.

Method	CATH 4.2 (test split)			TS50	TS500	Trainable/Total
	Short	Single-chain	All			
ProGen2-small [45]	13.46	12.68	11.70	12.68	9.49	-
InstructPLM-small	11.02	10.47	10.28	10.38	8.67	5.9M/306M(2.0%)
ProGen2-base [45]	9.12	10.91	8.33	8.08	5.81	-
InstructPLM-base	8.17	9.30	7.03	7.03	5.70	13.7M/927M(1.5%)
ProGen2-large [45]	8.33	9.48	7.77	7.46	5.70	-
InstructPLM-large	7.10	7.10	5.75	5.26	4.62	34.6M/2.9B(1.2%)
ProGen2-xlarge [45]	5.47	6.11	5.05	5.05	3.82	-
InstructPLM-xlarge	3.22	3.17	2.68	2.29	2.42	89.1M/6.6B(1.3%)

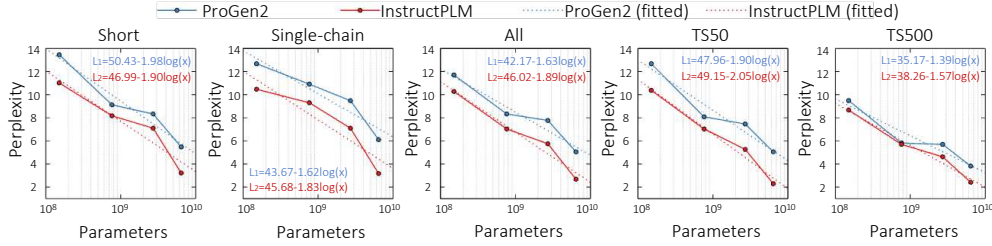


Fig. 4 Scaling law of InstructPLM w.r.t. perplexity on CATH 4.2 and TS50/500 respectively.

enzymes by sequence alignment and found that all of them have the same activate site
HIS179 which hit the UniRule PIRSR: PIRSR000102-1 [44] (See Supplementary Infor-
mation). An example of L-MDH structure predicted by ESMFold [39] is also shown in
Supplementary Information.

2.3 Ablation Studies and Discussion on Model Designs

2.3.1 Scaling Law of InstructPLM

Scaling laws [46], in the context of large language models, refer to the observation that
the next token prediction loss of these models scales as a power-law in terms of the
number of parameters, the amount of training data, or the computational resources
used during training. In this work, we mainly investigate the relationship between
the perplexity of InstructPLM and the model size of its ProGen2 pLM decoder. The
ProGen2 model family [45] includes autoregressive pLMs whose model size ranges
from 151M to 6.4B. We conduct an ablation study by enumerating pLM decoder in
InstructPLM from ProGen2-small (151M parameters), ProGen2-base (764M param-
eters), ProGen2-large (2.7B parameters), and ProGen2-xlarge (6.4B parameters). All
other model configurations are the same as Section 4.2. The experimental results are
shown in Table 2 and Figure 4. On all five datasets, both ProGen and InstructPLM
follow the scaling law of large language models — their language model loss can be

Table 3 Ablation study on protein backbone encoder: The perplexity of InstructPLM with different protein backbone encoders and different types of embeddings on CATH 4.2 test split, TS50, and TS500. The best performance is shown in **bold**.

Method	Dimension of Backbone Emb.	CATH 4.2 (test split)			TS50	TS500
		Short	Single-chain	All		
InstructPLM- PiFold	128	3.70	3.81	3.28	4.62	3.74
InstructPLM- ESM-IF	512	4.44	4.34	3.24	2.88	2.71
InstructPLM- MPNN	1152	3.22	3.17	2.68	2.29	2.42

Table 4 Ablation results on structure query length: the perplexity of CATH 4.2, TS50, and TS500 is obtained based on structure guidance of ProteinMPNN [7] embeddings. The amount of trainable parameters and total parameters are shown. The best performance is shown in **bold**.

#Queries	CATH 4.2 (test split)			TS50	TS500	Trainable/Total
	Short	Single-chain	All			
32	4.57	4.76	4.22	4.01	3.42	88.8M/6.6B(1.3%)
64	4.35	4.31	3.90	3.63	3.22	88.8M/6.6B(1.3%)
128	4.26	4.17	3.82	3.46	3.13	88.9M/6.6B(1.3%)
256	3.22	3.17	2.68	2.29	2.42	89.1M/6.6B(1.3%)
512	3.13	3.10	3.06	2.75	2.85	89.4M/6.6B(1.3%)

305 predicted using a power-law with respect to the model size. Compared to ProGen2,
 306 InstructPLM has achieved stable and consistent improvements across different model
 307 sizes and various datasets. This not only demonstrates the effectiveness of cross-
 308 modality alignment and instruction fine-tuning techniques we adopted in InstructPLM
 309 but also suggests that protein sequence design tasks can continuously benefit from the
 310 scaling of protein language models towards better performance.

311 2.3.2 Ablating Protein Backbone Encoder

312 We further ablate the usage of different protein backbone encoders in InstructPLM,
 313 including the encoders of ProteinMPNN [7], ESM-IF [8] and PiFold [36]. For Pro-
 314 teinMPNN, we concatenate the protein backbone representations encoded by nine
 315 released ProteinMPNN models¹, each with a dimension of 128, and finally obtain pro-
 316 tein backbone embedding with a dimension of $128 \times 9 = 1152$. For ESM-IF and PiFold,
 317 we adopt their backbone encoders which represent protein backbone structures to
 318 embeddings with dimensions of 512 and 128, respectively. Notably, although ESM-IF
 319 is originally trained on CATH 4.3, we present results of InstructPLM-**ESM-IF** trained
 320 on the CATH 4.2 dataset, mainly for a fair comparison. The experimental results are
 321 detailed in Table 3. Among all datasets, we observe that InstructPLM-**MPNN** yields
 322 the best performance, showing the robustness and effectiveness of ProteinMPNN.

2.3.3 Ablating the Number of Learnable Queries in Adapter

To probe the expression power of the protein structure-sequence adapter for sequence design, we conduct an ablation study on the number of learnable queries in the cross-attention of the adapter. We demonstrate the sequence design performance in terms of perplexity on the test split of CATH 4.2, TS50, and TS500 datasets. With other hyper-parameters the same as the training phase, the query length is chosen from 32, 64, 128, 256, and 512 respectively. The detailed performance of InstructPLM with different numbers of queries is shown in Table 4. InstructPLM achieves better perplexity as the increase of query length, illustrating how protein structure-sequence adapter compresses and provides meaningful instructions to the ProGen2 decoder. However, increasing the number of queries beyond 256 may lead to over-fitting. Although InstructPLM with 512 queries achieves better perplexity than InstructPLM with 256 queries on two subsets of CATH 4.2 (i.e., Short and Single-chain), we can observe significant performance drops on CATH 4.2 (All), TS50 and TS500. To utilize more robust models, we set the number of queries to 256 by default.

3 Conclusion

We propose a novel protein design framework called InstructPLM that leverages cross-modality alignment and instruction fine-tuning techniques. This innovative approach enables the generation of proteins that are not only variable in length but also diverse, adhering to specified structural guidelines. Significantly, InstructPLM’s ability to produce variable-length sequences introduces a novel dimension to protein design – mimicking the natural evolutionary processes of insertion and deletion. This capability addresses a previously unmet need, as prior methods are limited to generating sequences of fixed lengths. Furthermore, the proteins designed by our framework, evidenced through rigorous experimentation, exhibit the desired functional characteristics, demonstrating the efficacy and utility of our approach. To this end, we believe that InstructPLM will serve as a powerful toolkit that facilitates advancements in drug discovery, therapeutic proteins, and beyond.

4 Method

4.1 Model Architecture

For an input protein backbone, we aim to identify a suitable protein sequence S where $S = (s_1, s_2, \dots)$ which folds to the given structure. The overall framework of InstructPLM contains three main components, including the pre-trained protein backbone encoder, the protein structure-sequence adapter, and the pre-trained protein language model decoder. We present our models step by step.

¹<https://github.com/dauparas/ProteinMPNN>, including four vanilla models, two soluble models, and three C_α models.

4.1.1 Protein Backbone Encoder

We first obtain structure embedding $\mathbf{E} \in \mathbb{R}^{n \times D_{enc}}$ with a pre-trained structure encoder, where n is the number of protein residues and D_{enc} is the embedding dimension of the structure encoder. The embedding dimension D_{enc} can vary for different encoders. We utilize structure encoders from well-established protein sequence design methods, such as ProteinMPNN [7], ESM-IF [8], and PiFold [36]. To initialize these models, we make use of the officially released pre-trained weights. Specifically, for ProteinMPNN [7], we concatenate the embeddings from its nine models, which include four vanilla models, two soluble models, and three C_α models.

4.1.2 Protein Structure-Sequence Adapter

The protein structure-sequence adapter consists of four main components: linear projection layers for aligning structural embeddings to a latent interaction space, a dynamic positional encoding layer to capture residue-level connections for proteins, a cross-attention layer with a set of learnable queries $\mathbf{L} \in \mathbb{R}^{N \times D_{dec}}$ for knowledge transfer, where N is a tunable hyper-parameter and D_{dec} is determined by hidden dimension of pre-trained sequence decoder. Finally, we deploy an output projection layer that maps to language embeddings. To ensure the stability of module outputs, we further incorporate three normalization layers.

For an input structure embedding $\mathbf{E}$, the adapter will pad $\mathbf{E}$ with zeros to a fixed length n_s . We set n_s to 512 by default. We first incorporate positional information into the structure embedding $\mathbf{E}$ and learnable queries $\mathbf{L}$. For protein sequences with max length of n_s , the positional encoding ProtPE is defined for each dimension $i = 0, \dots, D_{dec}/2 - 1$ of the encoding at position $pos = 0, \dots, n_s - 1$ as:

$$\begin{aligned} \text{ProtPE}[pos, 2i] &= \sin\left(\frac{pos}{10000^{\frac{2i}{D}}}\right), \\ \text{ProtPE}[pos, 2i + 1] &= \cos\left(\frac{pos}{10000^{\frac{2i}{D}}}\right). \end{aligned} \quad (1)$$

The structure embedding $\mathbf{E}$ is projected to the latent interaction space before coupling with positional embedding:

$$\mathbf{E}' = \text{LayerNorm}(\mathbf{E}\mathbf{W}) + \text{ProtPE}, \quad (2)$$

where $\mathbf{W} \in \mathbb{R}^{D_{enc} \times D_{dec}}$. As the length of learnable queries $\mathbf{L}$ is inconsistent with the input structure length n_s , the positional information is injected into the queries by adding the positional feature embeddings with evenly distributed stride:

$$\mathbf{L}' = \text{LayerNorm}(\mathbf{L}) + \text{ProtPE}[:, : \text{stride}], \quad (3)$$

where the $\text{stride} = n_s/N$.

The cross-attention layer is formulated as a function that maps queries, keys, and values to outputs. We define keys and values according to the structural embeddings

in Equation 2 and query them with the learnable query as defined in Equation 3. To enhance the model’s capacity to attend to information from different sub-spaces, we employ a multi-head attention strategy [9], where the attention function is performed in parallel across A different attention heads. For each attention head $k \in \{1, 2, \dots, A\}$, the structural embeddings and queries are projected into the same dimension by linear projections:

$$\mathbf{Q}^k = \mathbf{L}'\mathbf{W}_Q^k, \quad \mathbf{K}^k = \mathbf{E}'\mathbf{W}_K^k, \quad \mathbf{V}^k = \mathbf{E}'\mathbf{W}_V^k, \quad (4)$$

where $\mathbf{W}_Q^k, \mathbf{W}_K^k, \mathbf{W}_V^k \in \mathbb{R}^{D_{dec} \times D_k}$ are parameter matrices for queries, keys, and values, respectively, and $D_k = D_{dec}/A$ represents the dimension of attention heads.

To transform the information from structural embeddings into the pre-defined queries, we obtain the interaction score based on inner products:

$$\mathbf{H}^k = \text{softmax} \left(\frac{\mathbf{Q}^k \mathbf{K}^{k\top}}{\sqrt{D_k}} \right) \mathbf{V}^k, \quad (5)$$

This operation computes the similarity between the queries and keys, which is then used to weight the values, effectively transferring the structural information to the outputs. The multi-head attention outputs are then concatenated across heads and linearly transformed to the final outputs of cross-attention:

$$\mathbf{O} = \text{Concat}(\mathbf{H}^1, \mathbf{H}^2, \dots, \mathbf{H}^A) \mathbf{W}_O, \quad (6)$$

where each $\mathbf{H}^k$ denotes independent outputs from a specific attention head, and $\mathbf{W}_O \in \mathbb{R}^{D_{dec} \times D_{dec}}$ is the parameter matrix for the final linear transformation.

Finally, a normalization layer and a linear projection layer $\mathbf{W}_p \in \mathbb{R}^{D_{dec} \times D_{dec}}$ are adopted to transform the attention output to the prompt $\mathbf{P}$ of the pLMs,

$$\mathbf{P} = \text{LayerNorm}(\mathbf{O}) \mathbf{W}_p. \quad (7)$$

Further details regarding the model’s weight initialization scheme and training hyperparameter settings are shown in Supplementary Information.

4.1.3 Protein Language Model Decoder

We utilize a pre-trained large protein language model as our sequence decoder. Specifically, we use ProGen2 [45] as our decoder, as ProGen2 is currently the largest and open-sourced auto-regressive protein language model. Given the structure prompt $\mathbf{P}$ obtained by previous steps, we concatenate the prompt $\mathbf{P}$ with the special token ‘1’ (following ProGen2, where ‘1’ indicates the N-terminal side of the sequence.) and send it to the sequence decoder model. The decoder generates the protein sequence from the conditional probability distribution of the next token in an auto-regressive manner,

$$s_t \sim \mathcal{P}(s_t | \mathbf{P}, '1', s_1, \dots, s_{t-1}). \quad (8)$$

417 For the fixed-length generation, we auto-regressively sample from the probability
 418 distribution in Equation 8 for n iterations. While for open-ended generation, we auto-
 419 regressively sample from the probability distribution in Equation 8 until encountering
 420 the C-terminal token, namely ‘2’. By integrating these components and techniques,
 421 InstructPLM efficiently processes protein structural embeddings on high-dimensional
 422 latent space and generate corresponding protein sequences.

423 4.2 Training

424 The InstructPLM is trained on the CATH 4.2 training set and validated on the
 425 CATH 4.2 validation set, no other data are included during training, suggesting the
 426 data efficiency of our methods. The training objective is to minimize the cross-entropy
 427 loss at the protein-token level, which measures the discrepancy between the predicted
 428 and ground-truth amino acid sequences.

$$\min_{\Theta} L(S, \Theta) = - \sum_{t=1}^{n+1} \log \mathcal{P}_{\Theta}(s_t | \mathbf{P}, '1', s_1, \dots, s_{t-1}), \quad (9)$$

429 where Θ represents the trainable parameters of the model, all from the protein
 430 structure-sequence adapter. The logarithm term computes the log-likelihood of the
 431 amino acid s_t given the protein structure prompt $\mathbf{P}$ and the preceding sequence tokens
 432 ‘1’, $s_1, \dots, s_{t-1}$. In addition, we also minimize the loss of the special C-terminal side
 433 token ‘2’ to ensure InstructPLM can stop properly at the end of sequences. Thus we
 434 append the special token ‘2’ at the end, i.e., $s_{n+1} = '2'$.

435 We use Adam [47] optimizer with a maximum learning rate of $1e^{-4}$, and a constant
 436 learning rate warm-up is applied for the first 5,000 steps to stabilize the learning pro-
 437 cess. The training procedure is performed on four NVIDIA A100 GPUs, with a batch
 438 size of 16 on each GPU, and using BF16 precision, enabling efficient parallel processing
 439 and reduced training time. The whole training configuration is summarized in Supple-
 440 mentary Information. By leveraging this training methodology, InstructPLM learns to
 441 effectively capture the relationship between protein structures and sequences, enabling
 442 accurate generation of sequences from given structural information. The frozen pre-
 443 trained ProGen2 provides a solid foundation, while the adapter focuses on learning
 444 the cross-modality alignment, offering a powerful solution to protein sequence design.

445 4.3 Protein Sequence Generation

446 For the distribution $\mathcal{P}_{\Theta}(S | \mathbf{P})$ learned by the InstructPLM, we use nucleus sam-
 447 pling [48] and temperature scaling to generate sequences. We first employ a
 448 temperature parameter τ to modulate the sharpness of the distribution,

$$\mathcal{P}'_{\Theta}(s_t | \mathbf{P}, '1', s_1, \dots, s_{t-1}) = \frac{\mathcal{P}_{\Theta}(s_t | \mathbf{P}, '1', s_1, \dots, s_{t-1})^{1/\tau}}{\sum_j \mathcal{P}_{\Theta}(s_t = j | \mathbf{P}, '1', s_1, \dots, s_{t-1})^{1/\tau}}. \quad (10)$$

449 A lower temperature value causes the distribution to become sharper, reducing the
 450 diversity of designed protein sequences. Then, nucleus sampling is adopted for truncat-
 451 ing the distribution to retain a subset (nucleus) that cumulatively covers a probability
 452 mass threshold specified by a hyperparameter top-p p . This method mitigates issues
 453 associated with other sampling approaches, such as the high randomness of pure sam-
 454 pling or the deterministic nature of greedy and beam search strategies. Formally, for
 455 the next token s_t over the vocabulary V , the smallest set V' of tokens is chosen by:

$$\sum_{s_t \in V'} \mathcal{P}'_{\Theta}(s_t | \mathbf{P}, '1', s_1, \dots, s_{t-1}) \geq p, \quad (11)$$

456 Finally, we sample new tokens from V' based on $\mathcal{P}'$.

457 In the context of generating fixed-length protein sequences, we stop the generation
 458 process when the InstructPLM attains the pre-defined length. When InstructPLM
 459 predicts the C-terminal side token before meeting the pre-defined length, we modify the
 460 token probability by setting the probability of the C-terminal side to 0 and continuing
 461 the generation process.

462 In the recovery rate (Table 1), and DEDAL (Figure 2(A)) benchmark, we first
 463 reproduce the result of three baselines (ProteinMPNN, PiFId, ESM-IF) utilizing the
 464 official codebase with temperature $\tau = 0.15$. We generate 20 sequences for each
 465 structure and report the mean recovery rate of 20 sequences as the final result. For
 466 InstructPLM, we adopt the fix-length generation for the recovery rate benchmark with
 467 the same temperature and a top-p of 0.9. In terms of the DEDAL benchmark, we let
 468 InstructPLM generate open-ended sequences with the same setting ($\tau = 0.15, p = 0.9$)
 469 for fair comparison.

470 When designing functional proteins, for a comprehensive exploration of the
 471 sequence space, we scale up our generation process to produce 10,000 sequences with
 472 a temperature of 0.8 and a top-p of 0.9. This extensive generation process facili-
 473 tates a robust evaluation of the InstructPLM’s ability to produce a wide array of
 474 plausible protein sequences conditioned on structural data, thereby enabling a more
 475 thorough assessment of the model’s performance and utility in computational biology
 476 and protein engineering tasks.

477 4.4 Experiment Details of Functional Proteins

478 We evaluate InstructPLM on two different proteins. PETase, a recently identified
 479 enzyme with the capability to degrade polyethylene terephthalate (PET), a common
 480 plastic, poses a test of InstructPLM’s ability to improve upon naturally occurring
 481 proteins. To further validate the generality and applicability of InstructPLM, we also
 482 investigate the designed sequences on L-MDH. L-MDH (Uniprot: A0A319AA41) is an
 483 important enzyme involved in the citric acid cycle, represents a fundamental metabolic
 484 process, and the absence of its definitive structure necessitates predictive modeling,
 485 testing the algorithm’s ability to function in the absence of solid experimental data.
 486 These proteins are chosen based on their varied structures and biological functions,
 487 presenting diverse challenges for InstructPLM and allowing us to probe the algorithm’s
 488 versatility and robustness.

489 4.4.1 Enzyme Expression and Purification of PETase

490 The recombinant bacteria were inoculated into LB liquid medium (100 µg/mL ampicillin) and grown overnight at 37°C/200 r.p.m. for 14 h. The 2% cell cultures were
491 transferred into 100 mL LB liquid medium (100 µg/mL ampicillin) and were shaken at
492 37 °C /200 r.p.m. for 3 h. When the optical density at 600 nm (OD_{600}) reached 0.6-0.8,
493 isopropyl-1-thio- β -D-galactopyranoside (IPTG) was added to final concentrations of
494 0.1 mM and grown overnight at 20°C/200 r.p.m. for 24 h. Subsequently, supernatants
495 were collected by centrifugation ($8000 \times g$ for 10 min at 4 °C), and resuspended in 50
496 mM Tris-HCl buffer (pH 7.5) after washing twice with buffer (pH 7.5). The washed
497 cells with 1 mg/mL lysozyme were disrupted by sonication (3-sec pulse on, 5-sec pulse
498 off, AMP 35%). Cell debris was removed by centrifugation at 4 °C, $8000 \times g$ for 1 h,
499 and the supernatants were filtrated by a 0.22 µm filter (Choice filter, Thermo Scientific). To obtain the purified enzymes, the supernatant, the samples were then applied
500 to a 5 µg/mL HisTrap HP column (GE Healthcare). After washing unbound proteins
501 (20 mM imidazole), the target protein was collected by 250 mM imidazole. The concentration of protein was determined using a protein assay kit (BCA Protein Assay
502 Kit; Genstar, Beijing, China) with BSA as the standard and the purity of each protein
503 was checked by SDS-PAGE analysis.
504
505
506

507 4.4.2 PET Depolymerization Assay

508 To evaluate the variants' activities, the amorphous gf-PET film (Goodfellow, 1 mm
509 thickness, $\varnothing 6$ mm, roughly 67 mg) was soaked in 1950 mL of 50 mM Tris-HCl buffer
510 (pH 8.5) and 50 µL of crude enzyme was added every 24 hours at of 40 °C. The reactions
511 were terminated by heat treatment (100 °C, 10 min). The supernatant obtained by
512 centrifugation ($8,000 \times g$ for 5 min) was then analyzed by high-performance liquid
513 chromatography (HPLC, Agilent 1200 and Ultimate 3000 UHPLC systems), equipped
514 with a Welch Ultimate XBC18 column (4.6×250 mm, 5 µm, Welch Materials, Inc.,
515 Shanghai, China) to quantify PET monomers released from PET depolymerization.
516 The mobile phase was 40% methanol with 0.12% acetic acid (pH 2.5) at a flow rate
517 of 0.7 mL/min, and the effluent was monitored at a wavelength of 240 nm.

518 4.4.3 Enzyme Expression and Purification of L-MDH

519 The E. coli DH5 α was used for plasmid construction, and E. coli BL21 (DE3) was
520 applied for gene expression. The gene was synthesized into the pET28 expression
521 plasmid. All BL21 (DE3) strains containing expression plasmids were cultivated in
522 Luria-Bertani (LB) liquid medium (5 g/L yeast extract, 10 g/L tryptone, 10 g/L
523 NaCl). And were inoculated at 37 °C with constant shaking at 220 r.p.m. Terrific
524 Broth (TB) medium, 24 g/L yeast extract, 12 g/L peptone, 4 mL glycerol, 2.31 g/L
525 KH₂PO₄, and 12.54 g/L K₂HPO₄, was the fermentation medium of the engineered
526 strains. The medium was supplemented with 50 µg/mL kanamycin (kan) according
527 to the screening markers carried by expression plasmids. Shake flask fermentation
528 was performed as follows: the engineered strains constructed with different expression
529 plasmids were grown in 5 mL of LB liquid medium overnight at 37 °C. Then, 50 µL
530 of the seed liquid was used to inoculate in 5 mL of TB medium. Cells were cultured

531 at 37 °C until OD_{600} reached up to 2-4, then Isopropyl β -D-1-thiogalactopyranoside
532 (IPTG) was added to a final concentration of 0.3 mM. The fermentation was allowed
533 to continue at 28 °C for an additional 10 h.

534 4.4.4 L-MDH Activity Assay

535 For L-MDH activity, 1 mL of fermentation broth was collected and centrifuged at
536 $15000 \times g$ for 3 min to obtain cell precipitation at 4 °C. The cell precipitates were
537 then resuspended in 1ml of phosphate-buffered saline (PBS). The cell suspensions were
538 lysed by sonication on ice: 30% power, working 5 s, interval 5 s for 10 min, to obtain
539 the MDH crude enzyme solution. The total reaction system is 200 μ L, including 50
540 mM PBS (176 μ L), 200 mM oxobutanedioic acid (2 μ L), 200 mM NADH+, and 20 μ L
541 dilution of crude enzyme solution. Then the change of absorbance at 340 nm was
542 detected and read absorbance data every 10s within 3 min. The enzyme activity was
543 calculated according to the following formula: $U/min = [\Delta A/min] \times [1/\epsilon] \times 1000$
544 ($\epsilon = 6.22 \text{ L}/(\text{mmol}/\text{cm})$).

545 Data Availability

546 All relevant data used to create training and evaluation inputs are available from public
547 sources. CATH dataset (v4.2) is available at <https://people.csail.mit.edu/ingraham/graph-protein-design/data/>, TS50 and TS500 dataset are available at <https://zenodo.org/records/6650679#.ZDJJNhVByhY> [49]. The structure of FAST-PETase [50] is
550 available in the PDB with accession numbers 7SH6. For L-MDH, the structure is
551 available at UniProt (<https://www.uniprot.org/uniprotkb/A0A319AA41/entry/>).

552 Code Availability

553 We used ESM-Fold [39] from fair-esm v2.0.0 (<https://pypi.org/project/fair-esm/2.0.0/>) for structure prediction. TM-Score calculation used DeepScore v1.12 (<https://github.com/realbigws/DeepAlign>). Other data analysis used Python v3.11.8 (<https://www.python.org/>), NumPy v1.24.3 (<https://numpy.org>), Matplotlib v3.7.1 (<https://matplotlib.org/>), seaborn v0.13.2 (<https://github.com/mwaskom/seaborn>) and
558 Veusz v3.6.2 (<https://github.com/veusz/veusz>). ProteinMPNN structure embed-
559 ding and benchmarking were done with a customized code (<https://github.com/chq1155/ProteinMPNN>). ESM-IF (<https://github.com/facebookresearch/esm>) and
561 PiFold (<https://github.com/A4Bio/PiFold>) benchmarking were done with official
562 code. Structure visualizations were created in Mol* [51] (<https://github.com/molstar/molstar>). Source code for the InstructPLM and trained weights are
564 available under an open-source license at <https://huggingface.co/InstructPLM/MPNN-ProGen2-xlarge-CATH42>. Inference and generation scripts are available under
566 an open-source license at <https://github.com/Eikor/InstructPLM>. Training code for
567 InstructPLM is not provided.

References

- [1] Kortemme, T. De novo protein design—from new structures to programmable functions. Cell **187**, 526–544 (2024).
- [2] Tallorin, L. et al. Discovering de novo peptide substrates for enzymes using machine learning. Nature communications **9**, 5253 (2018).
- [3] Schauperl, M. & Denny, R. A. Ai-based protein structure prediction in drug discovery: impacts and challenges. Journal of Chemical Information and Modeling **62**, 3142–3156 (2022).
- [4] de Haas, R. J. et al. Rapid and automated design of two-component protein nanomaterials using proteinmpnn. bioRxiv 2023–08 (2023).
- [5] Śledź, P. & Cafisch, A. Protein structure-based drug design: from docking to molecular dynamics. Current opinion in structural biology **48**, 93–102 (2018).
- [6] Defresne, M., Barbe, S. & Schiex, T. Protein design with deep learning. International Journal of Molecular Sciences **22**, 11741 (2021).
- [7] Dauparas, J. et al. Robust deep learning-based protein sequence design using proteinmpnn. Science **378**, 49–56 (2022).
- [8] Hsu, C. et al. Learning inverse folding from millions of predicted structures. International Conference on Machine Learning 8946–8970 (2022).
- [9] Vaswani, A. et al. Attention is all you need. Advances in neural information processing systems **30** (2017).
- [10] Sumida, K. H. et al. Improving protein expression, stability, and function with proteinmpnn. Journal of the American Chemical Society (2024).
- [11] Bennett, N. R. et al. Improving de novo protein binder design with deep learning. Nature Communications **14**, 2625 (2023).
- [12] Li, J., Li, D., Savarese, S. & Hoi, S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models (2023).
- [13] Bai, J. et al. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. arXiv preprint arXiv:2308.12966 (2023).
- [14] Alayrac, J.-B. et al. Flamingo: a visual language model for few-shot learning. Advances in Neural Information Processing Systems **35**, 23716–23736 (2022).
- [15] Liu, H., Li, C., Wu, Q. & Lee, Y. J. Visual instruction tuning. Advances in neural information processing systems **36** (2024).

- [16] Zhu, D., Chen, J., Shen, X., Li, X. & Elhoseiny, M. Minigpt-4: Enhancing vision-language understanding with advanced large language models. arXiv preprint arXiv:2304.10592 (2023).
- [17] Achiam, J. et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023).
- [18] Rives, A. et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proceedings of the National Academy of Sciences **118**, e2016239118 (2021).
- [19] Ferruz, N., Schmidt, S. & Höcker, B. Protgpt2 is a deep unsupervised language model for protein design. Nature communications **13**, 4348 (2022).
- [20] Madani, A. et al. Progen: Language modeling for protein generation. bioRxiv 2020-03 (2020).
- [21] Mao, W., Zhu, M., Chen, H. & Shen, C. Modeling protein structure using geometric vector field networks. bioRxiv 2023-05 (2023).
- [22] Zheng, Z. et al. Structure-informed language models are protein designers. bioRxiv 2023-02 (2023).
- [23] Verkuil, R. et al. Language models generalize beyond natural proteins. bioRxiv 2022-12 (2022).
- [24] Hie, B. L. et al. Efficient evolution of human antibodies from general protein language models. Nature Biotechnology (2023).
- [25] Sanh, V. et al. Multitask prompted training enables zero-shot task generalization. arXiv preprint arXiv:2110.08207 (2021).
- [26] Chung, H. W. et al. Scaling instruction-finetuned language models. arXiv preprint arXiv:2210.11416 (2022).
- [27] Ouyang, L. et al. Training language models to follow instructions with human feedback. Advances in Neural Information Processing Systems **35**, 27730–27744 (2022).
- [28] Rafailov, R. et al. Direct preference optimization: Your language model is secretly a reward model. Advances in Neural Information Processing Systems **36** (2024).
- [29] Eisenstein, M. Seven technologies to watch in 2024. Nature **625**, 844–848 (2024).
- [30] Jumper, J. et al. Highly accurate protein structure prediction with alphafold. Nature **596**, 583–589 (2021).
- [31] Zhang, Y. & Skolnick, J. Tm-align: a protein structure alignment algorithm based on the tm-score. Nucleic acids research **33**, 2302–2309 (2005).

- [32] Ingraham, J., Garg, V., Barzilay, R. & Jaakkola, T. Generative models for graph-based protein design. Advances in neural information processing systems **32** (2019).
- [33] Tan, C., Gao, Z., Xia, J., Hu, B. & Li, S. Z. Global-context aware generative protein design. ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 1–5 (2023).
- [34] Jing, B., Eismann, S., Suriana, P., Townshend, R. J. & Dror, R. Learning from protein structure with geometric vector perceptrons. arXiv preprint arXiv:2009.01411 (2020).
- [35] Gao, Z., Tan, C. & Li, S. Z. Alphadesign: A graph protein design method and benchmark on alphafolddb. arXiv preprint arXiv:2202.01079 (2022).
- [36] Gao, Z., Tan, C. & Li, S. Z. Pifold: Toward effective and efficient protein inverse folding (2022).
- [37] Orengo, C. A. et al. Cath—a hierarchic classification of protein domain structures. Structure **5**, 1093–1109 (1997).
- [38] Llinares-López, F., Berthet, Q., Blondel, M., Teboul, O. & Vert, J.-P. Deep embedding and alignment of protein sequences. Nature Methods **20**, 104–111 (2023).
- [39] Lin, Z. et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. Science **379**, 1123–1130 (2023).
- [40] Ma, J. & Wang, S. Algorithms, applications, and challenges of protein structure alignment. Advances in Protein Chemistry and Structural Biology **94**, 121–175 (2014).
- [41] Wang, S., Ma, J., Peng, J. & Xu, J. Protein structure alignment beyond spatial proximity. Scientific reports **3**, 1448 (2013).
- [42] Han, X. et al. Structural insight into catalytic mechanism of pet hydrolase. Nature communications **8**, 2106 (2017).
- [43] Ingraham, J. B. et al. Illuminating protein space with a programmable generative model. Nature 1–9 (2023).
- [44] MacDougall, A. et al. Unirule: a unified rule resource for automatic annotation in the uniprot knowledgebase. Bioinformatics **36**, 4643–4648 (2020).
- [45] Nijkamp, E., Ruffolo, J. A., Weinstein, E. N., Naik, N. & Madani, A. Progen2: exploring the boundaries of protein language models. Cell systems **14**, 968–978 (2023).

- 668 [46] Kaplan, J. et al. Scaling laws for neural language models.
669 arXiv preprint arXiv:2001.08361 (2020).
- 670 [47] Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization.
671 arXiv preprint arXiv:1412.6980 (2014).
- 672 [48] Holtzman, A., Buys, J., Du, L., Forbes, M. & Choi, Y. The curious case of neural
673 text degeneration (2019).
- 674 [49] Minami, S. Dataset for gcnDesign (2022). URL [https://doi.org/10.5281/zenodo.](https://doi.org/10.5281/zenodo.6650679)
675 [6650679](https://doi.org/10.5281/zenodo.6650679).
- 676 [50] Lu, H. et al. Machine learning-aided engineering of hydrolases for PET depolymer-
677 ization. Nature **604**, 662–667 (2022).
- 678 [51] Sehnal, D. et al. Mol* viewer: modern web app for 3D visualization and analysis
679 of large biomolecular structures. Nucleic acids research **49**, W431–W437 (2021).

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- [InstructPLMSI.pdf](#)