# BRIDGING PROTEIN STRUCTURE TO SEQUENCE VIA LOCAL STRUCTURE FOR INVERSE FOLDING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

The design of protein sequences based on given structures, known as inverse folding, has important applications in protein engineering. Protein structures are inherently hierarchical, composed of local structures (e.g., $\alpha$-helices and $\beta$-sheets) connected by loops and coils. However, most existing methods treat inverse folding as a direct 3D structure to 1D sequence task, ignoring this crucial hierarchical information embedded in local structures. In this work, we propose Hier-IF, a controllable inverse folding model that explicitly incorporates structural hierarchy. Hier-IF reformulates the task as a "Tertiary Structure (TS) to Local Structure (LS) to Sequence (Seq)" process by first generating the sequence tokens corresponding to local structures and then building the connecting loops and coils. We introduce classifier-free guidance for controllable hierarchical generation and employ a bidirectional structure-sequence reconstruction loss during the training process. In the sampling process, we design a remask strategy that enables controllable generation following the structural hierarchy. When evaluating Hier-IF across multiple datasets, it surpasses other baselines and achieves high structural fidelity in local structures. Visualizations on generation results and ablation studies in different experimental settings further validate the effectiveness of our approach and provide interpretability in protein hierarchical inverse folding.

## 1 INTRODUCTION

Protein sequences, as linear chains of amino acids, play a pivotal role in determining the structures and functions of cells and organisms. The design of protein sequences which can fold into specific structures is one of the core challenges in bio-engineering. Given a protein backbone structure, finding the corresponding sequence is called inverse folding. This task has substantial biological and practical significance (e.g., rational protein design (Street & Mayo, 1999), enzyme engineering (Fersht et al., 1992) and antibody design (Dreyer et al., 2023)). In nature, diverse sequences can fold into structurally similar conformations, revealing the redundancy and plasticity in the sequence–structure mapping. Understanding this inverse relationship is crucial for protein engineering, facilitating to design new proteins with desired functions, such as binding specificity, enzymatic activity, or stability under industrial conditions. Furthermore, inverse folding can assist in validating whether a predicted or synthetic structure is biophysically plausible by checking if any natural-like sequence can adopt it.

Deep learning models have made a significant progress in this area(Dauparas et al., 2022; Hsu et al., 2022; Gao et al., 2022a). These approaches formulate inverse folding as a structure-to-sequence ("str2seq") task. By learning from tons of corresponding pairs of protein structures and sequences, such models can capture the underlying relationship and generate sequences conditioned on a given protein structure. However, such "str2seq" methods oversimplify the real chemical and biological processes. The intermediate process of this transformation has been ignored. Proteins exhibit a hierarchical structural organization that is crucial for their biological functions. Proteins include some local structures, such as alpha-helices, beta-sheets and loops (or coils)(Kuhlman & Bradley, 2019; Martí-Renom et al., 2000). These secondary elements further assemble into a unique tertiary structure, representing the overall 3D conformation of a single polypeptide chain. The tertiary structure often involves intricate spatial arrangements of alpha-helices and beta-sheets connected by flexible loops, resulting in a stable fold capable of performing specific functions, such as molecular recognition or catalysis. In structural biology (Crick, 1958; Koga et al., 2012), when synthesizing a protein, local structures such as alpha-helix or beta-sheet in the structure are usually synthesized first, followed by

the assembly of loops to complete structure. Therefore, effectively modeling these local structures is crucial, yet this aspect is underexplored in current deep learning approaches.

Approaches to inverse folding can be broadly divide into two strategies. One class of methods performs "str2seq" transformation using a structure encoder and a sequence decoder. While protein sequences inherently carry a lot of evolutionary and homologous information (Whisstock & Lesk, 2003; Jumper et al., 2021), these methods may not learn and utilize such signals , because they usually use a GNN-based model to capture the structure information and lack the learning of sequence information. The other approach is to adopt protein language model trained on large-scale protein sequence datasets. These methods consider the given structure as a condition to generate desired sequence. However, in such frameworks, the structural information and the sequence information are not treated symmetrically, and the fine-grained structural information is not be fully utilized. The relationship between the structure and the corresponding sequence is often modeled implicitly.

To address the aforementioned challenges, we propose **Hier-IF**, a biologically grounded pipeline for inverse folding that formulates the generation as a **Tertiary Structure (TS) to Local Structure (LS) to Sequence (Seq)** process. In contrast to existing paradigms that encode backbone geometry as a single monolithic constraint, Hier-IF explicitly models the hierarchical dependencies between global topology and local structure. We first encode protein structures into discrete tokens and employ a unified masked discrete diffusion framework to model the joint probability of structure and sequence. To enforce hierarchical consistency, we utilize CATH (Orengo et al., 1997) classifications as supervisory signals and introduce a time-aware masking schedule that prioritizes the generation of stable secondary elements (helices/strands) before flexible regions (loops). Furthermore, to ensure the synthesized sequences fold back into the desired structures, we introduce a bidirectional structure–sequence reconstruction loss, coupling a forward-folding objective with the inverse-folding likelihood. This streamlined design allows Hier-IF to achieve competitive performance with complex refinement-based systems while offering superior interpretability and control over the generation process.

**We summarize our key contributions as follows**.

- We propose an innovation Tertiary Structure (TS) to Local Structure (LS) to Sequence (Seq) protein inverse folding pipeline, which explicitly models hierarchical protein information.
- We introduce a bidirectional structure–sequence reconstruction loss to jointly model sequence–structure consistency and propose a time-aware mask to control the generation order. Furthermore, we design a controllable hierarchical sampling strategy that combines classifier-free guidance and a structure-aware remask mechanism to ensure generation follows the structural hierarchy.
- Hier-IF surpasses other baselines by evaluating it on various inverse folding settings. The visualization and ablation study validate the effectiveness of our approach and provide interpretability in protein hierarchical inverse folding.

## 2 Related Work

### 2.1 Protein Language model

With the advancement of large-scale protein sequence datasets, numerous protein language models—drawing inspiration from natural language processing (NLP)—have demonstrated remarkable potential. Models such as ProtTrans (Elnaggar et al., 2021) and TAPE (Rao et al., 2019) capture the information embedded in protein sequences by framing protein-related tasks as classic language modeling problems, such as masked language modeling and text-to-text generation. In parallel, the evolutionary information contained within protein sequences provides powerful inductive biases that facilitate the learning of biologically meaningful representations. For example, models like MSA-Transformer (Rao et al., 2021) leverage multiple sequence alignments (MSAs), leading to more effective learning. The ESM-series models (Rives et al., 2021; Lin et al., 2023; Hayes et al., 2025) utilize large-scale language modeling over hundreds of millions of protein sequences, implicitly capturing the evolutionary information and enabling the extraction of robust sequence embeddings that strongly correlate with protein structures and functions. Furthermore, with the development of generative models, methods such as ProGen (Madani et al., 2023) and ProtGPT2 (Ferruz et al., 2022)

have demonstrated the capability to not only learn robust representations but also generate desired protein sequences, making them applicable to a wide range of downstream tasks.

### 2.2 Inverse Folding

Based on given protein structure, i.e., backbone, designing the sequence is called inverse folding. Using a structural encoder and a sequence decoder is a straightforward yet effective strategy for structure-based protein design. This framework allows models to extract structural context and generate compatible amino acid sequences. ProteinMPNN (Dauparas et al., 2022) employs a message-passing neural network directly on residue coordinates to capture local geometric features, followed by an autoregressive decoder. In contrast, PiFold (Gao et al., 2022a) constructs a residue-level graph and uses a graph neural network to model both spatial and topological relationships, with a transformer-based decoder that better captures long-range dependencies. These differences reflect distinct design choices in how structural information is encoded and utilized for sequence generation. On the other hand, LM-Design (Zheng et al., 2023) and the DPLM series (Wang et al., 2024a;b) have effectively leveraged pretrained language models by introducing lightweight adapters. These methods insert small trainable modules into frozen pretrained models, enabling efficient fine-tuning while preserving the general knowledge encoded in the pretrained weights. This design allows for task-specific adaptation with minimal overhead, showing strong performance in structure-conditioned sequence design. Such approaches offer a promising and scalable paradigm for future research. KW-design proposes a refining method that considers multimodel knowledge as well as predictive confidence. Bridge-IF (Zhu et al., 2024) introduce Schrödinger bridge to handle the challenges in structure-sequence mapping. UniIF (Gao et al., 2024) proposes an unified molecule inverse folding. By effective representation of all types of molecules, it can achieve great performance in small molecules, proteins and RNAs. Furthermore, ProteinInvBench (Gao et al., 2023b) and ProteinBench (Ye et al., 2024) provide comprehensive benchmarks for inverse folding and related metrics. They systematically organize a range of evaluation tasks and define detailed metrics to assess model performance across different settings. These benchmarks offer a standardized framework for fair comparison and have become valuable resources for evaluating progress in structure-conditioned sequence generation.

## 3 Methods

### 3.1 Framework

We propose a "TS-LS-Seq" inverse folding framework, as illustrated in Figure 1. In Figure 1(a), we tokenize the protein backbone using a structure-aware encoder, effectively encoding the backbone as a "structure language". Specifically, we adopt the pretrained ESM-3 model as the model encoder. In Figure 1(b), we introduce a masked language model to learn the mapping from the tertiary structures to local substructures and then to the full sequence. For training, we design a bidirectional loss function, distinguishing our method from previous inverse folding approaches. To determine the generation order, we design a DSSP-based masking strategy, where the mask is directly generated by the model. Rather than optimizing solely on sequence reconstruction, we further perform forward folding on the generated sequence to recover its 3D structure, and jointly optimize both sequence and structure reconstruction losses. Additionally, we incorporate CATH classification labels to provide hierarchical structural guidance during generation, enabling the model to better capture domain-level structural organization. As shown in Figure 1(c), We utilize a trained mask generator to remask the generated sequence, guiding the model to generate local structural motifs (e.g., helices and strands) first. Through iterative generation and remasking, the complete sequence is progressively constructed. This mechanism ensures that connecting regions, such as loops and coils, are generated subsequently, once the local structures have been established.

### 3.2 Tokenization for Protein Structure

The first step of our model is to capture the information of proteins. Inspired by recent advances in structural vocabularies for protein representation learning (van Kempen et al., 2022; Su et al., 2023; Wang et al., 2024b), we use discrete token to represent protein tertiary structures. Tokenizing continuous data modalities into discrete representations (Van Den Oord et al., 2017) has attracted

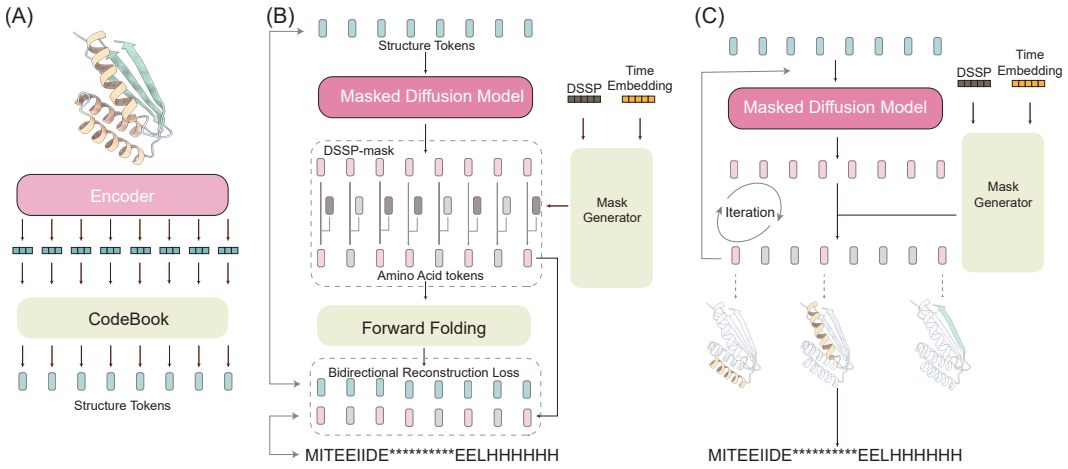


Figure 1: The Framework of Hier-IF

considerable attention across various fields. On the one hand, this approach can efficiently model the structural information and sequential order. On the other hand, it can be easily applied to powerful sequence-based models (e.g. transformer-based models).

In Hier-IF, we employ the ESM3 (Hayes et al., 2025) encoder as our tokenizer. Trained on vast amounts of data, ESM3 effectively captures rich protein information. Unlike previous ESM versions, ESM3 supports protein co-design and allows the sequence input to be set as None to directly obtain structural representations. Therefore, we consider ESM3 a powerful and versatile tool. By inputting the backbone, we can extract the corresponding structure tokens. These tokens are obtained from the pretrained ESM3 model, providing high-quality representations that encapsulate extensive protein information. This approach enables us to convert protein structures in datasets into meaningful tokens, which can then serve as inputs for subsequent training stages.

### 3.3 Conditional Masked Diffusion Language Model with Mask Generator

**Conditional Masked Diffusion Language Mode**. The discrete diffusion models can be generally defined by a sequential process of progressive noisy discrete variables $z_t \in \mathcal{V}$ from the categorical variable $z_0 \in \mathcal{V}$. Masked diffusion (Austin et al., 2021; Lou et al., 2023; Shi et al., 2024) represents a special case in which the transition includes an "absorbing state", denoted as [MASK]. In this formulation, the stationary distribution assigns all probability mass to the unique special token [MASK], such that $P(z = \text{[MASK]}) = 1$ and $P(z \neq \text{[MASK]}) = 0$. Following Lu et al. (2024), we define $\mathbf{p}_M \in \{0,1\}^{|\mathcal{V}|}$ ($\mathcal{V} = \mathcal{V} \cup \{\text{[MASK]}\}$) as the one-hot vector representing [MASK]. In masked diffusion, the stochastic forward process maps $z_0$ to [MASK] and remains in this state thereafter (i.e., "absorbing"). Conversely, the reverse process gradually unmasks (denoises) the [MASK] token to produce the data sample $z_0$, where $s < t$:

$$q(z_s|z_t, z_0) = \text{Cat}(z_s; [\beta(s,t) + (1 - \lambda_M(z_t))(1 - \beta(s,t))]z_t + \lambda_M(z_t)(1 - \beta(s,t))z_0), \quad (1)$$

where $\beta(s,t) = \frac{1-\alpha(s)}{1-\alpha(t)}$ and $\lambda_M(z_t) = \langle \mathbf{p}_M, z_t \rangle$. Equation 1 implies that when $z_t \neq$ [MASK], the backward process copies the unmasked token directly, i.e., $z_s \leftarrow z_t$, corresponding to the categorical distribution $q(z_s \mid z_t, z_0) = \text{Cat}(z_s; z_t)$. Otherwise, when $z_t =$ [MASK], the posterior is given by a convex interpolation between $\mathbf{p}_M$ and $z_0$. The posterior $q(z_s \mid z_t, z_0)$ can be approximated via reparameterization as $p_\theta(z_s \mid z_t) = q\big(z_s \mid z_t, u_\theta(t, z_t)\big)$, where the neural network $u_\theta \in \Delta^{|\mathcal{V}|}$ outputs a probability vector confined to the simplex $\Delta^{|\mathcal{V}|}$. While unconditional protein sequence generation lacks structural constraints and positional alignment, inverse folding is well-defined within discrete diffusion models due to its explicit conditioning on the input backbone. This structure-based conditioning ensures a direct mapping between output residues and backbone positions, enabling generation maintaining a fixed-length context.

For inverse folding generation, we consider a conditional masked diffusion. The given backbone corresponding tokens can be regarded as a condition $c$. Our goal is to generated the protein sequences(e.g. Amino acid type sequences) by a conditional posterior $q(z_s \mid z_t, z_0; c)$.

**Time-Aware Generative Mask**. To guide the diffusion model to prioritize structured regions (e.g., helices and strands) early in the generation process, we propose a unified structure- and time-aware control mechanism that consists of (1) a soft mask generator that modulates the model outputs, and (2) a dynamic loss reweighting strategy that adjusts the training focus over time.

We introduce a multi-layer perceptron (MLP) that generates a soft mask vector $m \in [0,1]^L$ for each input sequence, where $L$ is the sequence length. The input to the MLP is a concatenation of the DSSP one-hot encoding and a positional embedding of the current diffusion timestep $t$. The DSSP (Define Secondary Structure of Proteins) algorithm is a standard method for assigning secondary structure to amino acid residues in protein structures based on hydrogen bonding patterns and backbone geometry. The mask is applied element-wise (position-wise) multiplicatively to the model's predicted logits $\hat{y}$:

$$\hat{y}' = m \odot \hat{y} \tag{2}$$

This modulation allows the model to focus more on structurally meaningful regions at appropriate timesteps.

To further reinforce the desired generation order, we define a per-position weight function $w_i(t)$ that adjusts the loss contribution based on the DSSP label and current timestep:

$$w_i(t) = \begin{cases} 1 + \alpha \left(1 - \frac{t}{T}\right), & \text{if DSSP}_i \in \{\text{H}, \text{E}\} \\ 1, & \text{if DSSP}_i = \text{C} \end{cases} \tag{3}$$

Here, $T$ is the total number of diffusion steps, and $\alpha$ is a hyperparameter controlling the early-stage emphasis on helices and strands. The final training loss is computed as a weighted average over per-position prediction losses $\mathcal{L}_i$ calculated from the masked logits $\hat{y}'_i$:

$$\mathcal{L}_{\text{final}} = \frac{1}{\sum_i w_i(t)} \sum_i w_i(t) \cdot \mathcal{L}_i(\hat{y}'_i). \tag{4}$$

This unified mechanism imposes an explicit structural prior over the generation process, ensuring that well-structured elements are synthesized early while coil regions are refined at later stages.

**Bidirectional Structure-Sequence Reconstruction Loss**. To improve consistency between sequences and structures in inverse folding, we propose a bidirectional reconstruction loss combining sequence and structural components:

$$\mathcal{L}_{\text{bi}} = \lambda_{\text{seq}} \cdot \text{CE}(\hat{S}, S_{\text{true}}) + \lambda_{\text{struct}} \cdot \mathcal{D}\big(f_{\text{fold}}(\hat{S}), X_{\text{true}}\big), \tag{5}$$

where $\hat{S}$ is the predicted sequence, $S_{\text{true}}$ the ground-truth sequence, $X_{\text{true}}$ the native backbone structure, $f_{\text{fold}}$ a pre-trained folding model mapping sequence to structure, CE the cross-entropy loss, and $\mathcal{D}$ a differentiable structural distance metric. The hyperparameters $\lambda_{\text{seq}}$ and $\lambda_{\text{struct}}$ balance the two terms. We set as$\lambda_{\text{seq}} : \lambda_{\text{struct}} = 0.8 : 0.2$. In our work. Besides, we employ the ESM-3 model as $f_{\text{fold}}$ to predict the structure from sequences.

**Fold-Class Conditional Guidance**. Fold class labels in CATH datasets describe protein structures at different levels of detail, and C-level labels include the information of local structure. so, awareness of this label can improve the capacity of models.Followed by different label combinations during training in Geffner et al. (2025), we train our model in the same way. Specifically, with $p = 0.5$ we drop all labels ($\{\emptyset, \emptyset, \emptyset\}_{\text{CAT}}$), with $p = 0.25$ we only show the C label ($\{C_x, \emptyset, \emptyset\}_{\text{CAT}}$), with $p = 0.15$ we drop only the T label ($\{C_x, A_x, \emptyset\}_{\text{CAT}}$), and with $p = 0.1$ we give the model all labels ($\{C_x, A_x, T_x\}_{\text{CAT}}$). Moreover, these methods enables classifier-free guidance (Ho & Salimans, 2022) for all possible levels during inference, combining the unconditional model prediction with any of the label-conditioned predictions .

The overall training loss of our mask diffusion model integrates structure- and time-aware weighting with bidirectional reconstruction loss:

$$\mathcal{L}_{\text{total}} = \underbrace{\frac{1}{\sum_i w_i(t)} \sum_i w_i(t) \cdot \mathcal{L}_{\text{pred}}\big(m_i \cdot \hat{y}_i, y_i\big)}_{\text{Mask diffusion loss with time-aware mask}} + \underbrace{\lambda_{\text{seq}} \cdot \text{CE}(\hat{S}, S_{\text{true}}) + \lambda_{\text{struct}} \cdot \mathcal{D}\big(f_{\text{fold}}(\hat{S}), X_{\text{true}}\big)}_{\text{Bi-directional reconstruction loss}}, \tag{6}$$

Here, $m_i$ is the soft mask generated by a neural network, $t$ is the current diffusion timestep, $T$ is the total number of steps, and other notations are as previously defined. The detailed derivation of the loss function and the specific training settings are provided in the Appendix.

### 3.4 Sampling and Remask with Trained Mask Generator

In the sampling stage of our masked diffusion model, we introduce a remask strategy guided by a trained mask generator. This generator produces a new binary mask at each timestep, which is then used to selectively remask part of the current generated state. The goal is to allow the model to revise uncertain or structurally important regions, while retaining confident predictions from previous steps.

Formally, let $\mathbf{x}_t \in \mathcal{V}^L$ denote the generated sequence or structure representation at diffusion step $t$, where $\mathcal{V}$ is the vocabulary and $L$ is the sequence length. The trained mask generator produces a soft mask and soft mask transforms to a binary hard mask $\hat{\mathbf{m}}_t \in \{0,1\}^L$, indicating which positions should be masked again: $\hat{\mathbf{m}}_t = \text{MaskGenerator}(t, \mathbf{s})$, where $\mathbf{s}$ denotes the DSSP-derived secondary structure annotation.

This process is repeated from $t = T$ down to $t = 1$, gradually refining the sample by re-evaluating a subset of tokens at each step based on the current state and structural priors.

**Algorithm 1** DSSP-Guided Remasked Sampling

**Require:** Total diffusion steps $T$, initial input $\mathbf{x}_T$, DSSP annotation $\mathbf{s}$
**Ensure:** Final generated result $\mathbf{x}_0$
1: **for** $t = T, T-1, \ldots, 1$ **do**
2: $\quad \mathbf{x}_t \leftarrow \text{Sample}(\mathbf{x}_{t+1}, c)$
3: $\quad \hat{\mathbf{m}}_t \leftarrow \text{MaskGenerator}(t, \mathbf{s})$
4: $\quad \mathbf{x}'_t \leftarrow \mathbf{x}_t \odot (1 - \hat{\mathbf{m}}_t) + [\text{MASK}] \odot \hat{\mathbf{m}}_t$
5: $\quad \mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}'_t, c)$
6: **end for**
7: **return** $\mathbf{x}_0$

This strategy enables controllable and structure-aware resampling, allowing the model to focus on refining regions of high uncertainty or structural importance at each generation step.

## 4 Experiments

In this section, we evaluate Hier-IF on a variety of benchmarks for the inverse folding task. We begin with experiments on protein sequence design using the CATH4.2 and CATH4.3 benchmarks. Then, we demonstrate that Hier-IF can generate protein sequences in a controllable manner with high structural fidelity. Extensive evaluations under diverse settings show that Hier-IF effectively captures hierarchical structural information and enables controllable sequence generation. Moreover, ablation studies and visualizations further illustrate the effectiveness and interpretability of the model.

### 4.1 Benchmarking Inverse Folding

**Objective & Setting** We demonstrate the effectiveness of Hier-IF on the widely used CATH (Orengo et al., 1997) dataset. To provide a comprehensive comparison, we conduct experiments on both CATH4.2 and CATH4.3. The CATH4.2 consists of 18024 proteins for training, 608 proteins for validation, and 1120 proteins for testing, following the same data splitting as Ingraham et al. (2019).

The CATH4.3 dataset includes 16153 structures for the training set, 1457 for the validation set, and 1797 for the test set, following the same data splitting as Hsu et al. (2022). To evaluate the generative quality, we report median recovery scores of the top-1 predicted sequences on short-chain, single-chain, and all-chain settings and perplexity.

**Baselines** We compare Hier-IF with recent graph models, including GraphTrans (Ingraham et al., 2019),GVP (Jing et al., 2020), AlphaDesign (Gao et al., 2022b), ESM-IF (Hsu et al., 2022),ProteinMPNN (Dauparas et al., 2022),PiFold (Gao et al., 2022a). The ESM-based baselines include LM-Design (Zheng et al., 2023) and KW-Design (Gao et al., 2023a). The diffusion-based baselines include Grade-IF (Yi et al., 2023) and Bridge-IF (Zhu et al., 2024).To demonstrate the flexibility of our proposed framework, we also report results for a variant of Hier-IF instantiated with FoldSeek as the backbone (in addition to our default ESM3).To provide a fair comparison with ESM-IF, our model train on the CATH4.3 dataset following the same data splitting as ESM-IF.

**Results & Analysis** From Table 1, we conclude that Hier-IF achieves state-of-the-art performance on different settings. Specifically, we observe the following: (1)Hier-IF can achieve the best recovery in short and single-chain settings on CATH4.2 and CATH4.3. On all chain setting, it can also achieve comparable results with KW-design which learn lot of knowledge by tuning with several pretrained model. (2) Hier-IF can achieve comparable performance with the KW-design with less cost and surpass other methods. (3) Compared with other methods without ESM, ESM-based model achieve obvious improvement in recovery and perplexity, which means that use the evolutionary information of protein sequence is crucial in inverse folding.

Table 1: Comparison on CATH 4.2 and 4.3. † denotes ESM utilization. Best in **bold**, second in underlined.

| Dataset | Model | Recovery (↑) | | | Perplexity (↓) | | |
|---|---|---|---|---|---|---|---|
| | | Short | Single | All | Short | Single | All |
| **CATH 4.2** | GraphTrans | 0.28 | 0.28 | 0.36 | 8.39 | 8.83 | 6.63 |
| | GVP | 0.31 | 0.29 | 0.39 | 7.23 | 7.84 | 5.36 |
| | PiFold | 0.40 | 0.39 | 0.52 | 6.04 | 6.31 | 4.55 |
| | AlphaDesign | 0.34 | 0.33 | 0.41 | 7.32 | 7.63 | 6.30 |
| | ProteinMPNN | 0.36 | 0.34 | 0.46 | 6.21 | 6.68 | 4.61 |
| | ESM-IF† | 0.31 | 0.39 | 0.38 | 8.18 | 6.33 | 6.44 |
| | LM-Design† | 0.38 | 0.42 | 0.56 | 6.77 | 6.46 | 4.52 |
| | KW-Design† | 0.45 | 0.45 | **0.61** | 5.48 | **5.16** | 4.02 |
| | GraDe-IF | 0.45 | 0.43 | 0.52 | 5.49 | 6.21 | 4.35 |
| | Bridge-IF | 0.44 | 0.45 | 0.58 | 5.68 | 5.27 | **3.38** |
| | Hier-IF (FoldSeek) | 0.43 | 0.44 | 0.56 | 6.01 | 5.83 | 4.44 |
| | Hier-IF† (ESM3) | **0.46** | **0.46** | 0.59 | **5.46** | 5.17 | 4.01 |
| **CATH 4.3** | GraphTrans | 0.30 | 0.34 | 0.34 | 8.37 | 8.83 | 6.61 |
| | GVP | 0.33 | 0.36 | 0.38 | 7.21 | 7.85 | 5.34 |
| | PiFold | 0.43 | 0.52 | 0.51 | 6.04 | 6.27 | 4.52 |
| | AlphaDesign | 0.43 | 0.43 | 0.42 | 7.32 | 7.60 | 6.28 |
| | ProteinMPNN | 0.38 | 0.44 | 0.44 | 6.21 | 6.61 | 4.61 |
| | ESM-IF† | 0.34 | 0.39 | 0.38 | 8.18 | 6.32 | 6.42 |
| | LM-Design† | 0.47 | 0.48 | 0.56 | 5.66 | 5.52 | 4.01 |
| | KW-Design† | 0.44 | 0.46 | **0.60** | 5.47 | 5.23 | **3.49** |
| | GraDe-IF | 0.46 | 0.44 | 0.54 | 5.50 | 6.13 | 4.29 |
| | Bridge-IF | 0.47 | 0.48 | 0.59 | 5.49 | 5.63 | 3.90 |
| | Hier-IF (FoldSeek) | 0.47 | 0.48 | 0.54 | 5.69 | 5.54 | 4.09 |
| | Hier-IF† (ESM3) | **0.48** | **0.49** | **0.60** | **5.37** | **5.22** | 3.88 |

### 4.2 HIER-IF CAN LEARN THE HIERARCHICAL INFORMATION

To further analyze the capacity of Hier-IF in capturing the hierarchical information, we examine the secondary structure distribution against proteins from CATH.

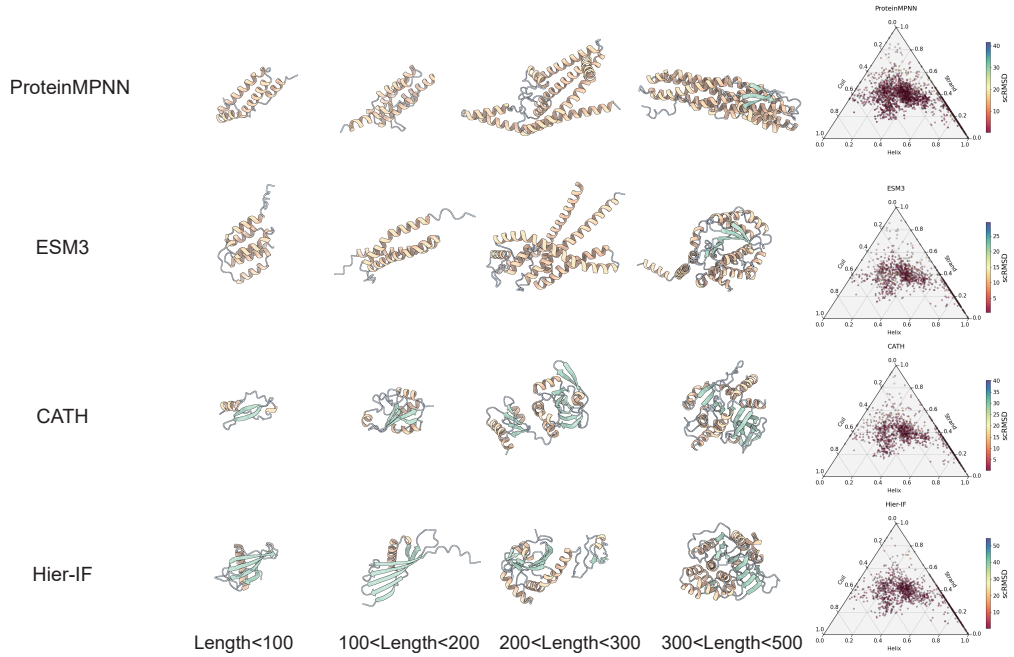


Figure 2: Sampling results of ProteinMPNN, ESM3, CATH and Hier-IF, as well as secondary structure distribution

**Proteins sampled by Hier-IF have secondary structures most similar to trained datasets**. As seen in Fig 2, proteinMPNN generate more helices and fewer strands and coils than natural proteins. ESM3 and Hier-IF show a little bias toward the proteins from CATH, but ESM3 tends to generate more strands and coils. Among the methods, Hier-IF can effectively capture hierarchical information and generate the most natural-like secondary structure proportions matching natural protein more. Proteins generated by ProteinMPNN contain a lot of helices and the issue becomes increasingly severe as the length increases. Proteins generated by While ESM3 can generate a certain number of strands and coils, there remains a noticeable gap compared to real natural proteins.

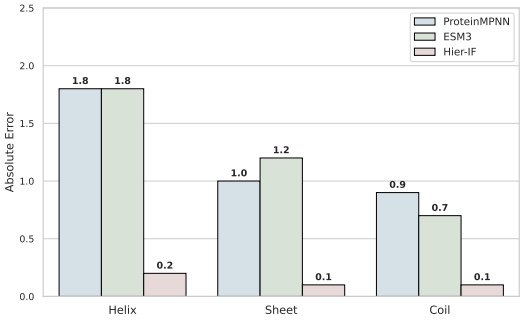


Figure 3: Secondary Structure Fidelity (Error vs. CATH).

Table 2: Comparison of secondary structure recovery measured by MAPE.

| Method | MAPE |
|---|---|
| ProteinMPNN | 7.7% |
| ESM3 | 7.6% |
| **Hier-IF (Ours)** | **1.0%** |

As illustrated in Figure 3, baseline methods like ProteinMPNN and ESM-3 exhibit specific structural biases, most notably an over-generation of helices at the expense of sheets. Hier-IF minimizes these specific deviations, maintaining a profile that is nearly indistinguishable from the natural baseline (CATH) across all secondary structure types. This component-wise fidelity is aggregated and quantitatively confirmed in Table 2. In terms of Mean Absolute Percentage Error (MAPE), Hier-IF achieves a remarkably low error of 1.0%. This represents an approximate 87% reduction in overall compositional error compared to baselines, demonstrating that our model captures the hierarchical statistics of protein structures with high precision.

### 4.3 HIERARCHY-GUIDED FOR CONTROLLABLE GENERATION

To address the helix over-generation bias, we applied C-level secondary structure guidance to Hier-IF, following the protocol of Geffner et al. (2025). Table 3 demonstrates the effectiveness of this approach: guiding towards the "Main-$\beta$" class dramatically increases $\beta$-sheet content (to 32.7%), effectively correcting the imbalance seen in unconditional generation. Crucially, this structural conditioning enhances designability, peaking at 95.2% in the "Mixed" class. We observe that this improvement is accompanied by a moderate reduction in diversity (FoldSeek scores). This reflects an expected trade-off: introducing class constraints narrows the conformational landscape, balancing specificity against variability. However, global structural fidelity remains uncompromised. The TM-scores remain stable across all settings, indicating that while Hier-IF becomes more specific under guidance, it retains the capacity to generate biologically plausible folds without structural degradation.

Table 3: Guiding Hier-IF in C-level Classes.

| Class | Des.(%) | Div. | | SS (%) | |
|---|---|---|---|---|---|
| | | FS | TM | $\alpha$ | $\beta$ |
| Uncond. | 92.1 | 0.53 | 0.84 | 38.6 | 23.7 |
| Main $\alpha$ | 92.6 | 0.26 | 0.90 | 58.9 | 7.2 |
| Main $\beta$ | 89.0 | 0.39 | 0.83 | 17.5 | 32.7 |
| Mix | 95.2 | 0.39 | 0.82 | 44.1 | 20.6 |

### 4.4 EVALUATION OF GENERALIZATION AND FOLDABILITY

To comprehensively assess the capabilities of Hier-IF beyond standard benchmarks, we conducted extensive experiments focusing on two critical aspects: the model's generalization to independent datasets and the structural validty of the designed sequences. Standard benchmarks often share distribution characteristics with the training set. To evaluate the robustness of Hier-IF on unseen protein distributions, we tested our model on the TS50 and TS500 datasets, which are widely used as independent test sets. As detailed in Appendix E, Table 5, Hier-IF demonstrates superior generalization capabilities. Specifically, on the larger TS500 dataset, our model achieves a recovery rate of 66.42%, outperforming state-of-the-art baselines. This indicates that the hierarchical information learned by our model transfers effectively to novel protein topologies.

High sequence recovery does not always guarantee that the generated sequences will fold into the desired structures. To verify the physical plausibility and *foldability* of our designs, we performed an *in silico* folding validation using OpenFold. We predicted the 3D structures of the generated sequences and calculated the TM-score and scRMSD against the ground truth backbones. The results in Table 4, show that Hier-IF achieves the highest TM-score (0.82) and ties for the best scRMSD (1.36 Å). This confirms that Hier-IF generates sequences that are not only statistically likely but also structurally consistent with the target geometries.

Table 4: **In silico folding validation.** We compare the structural consistency. **Bold** indicates the best.

| **Model** | **Rec.(↑)** | **TM(↑)** | **scRMSD(↓)** |
|---|---|---|---|
| ProteinMPNN | 0.46 | 0.80 | **1.36** |
| PiFold | 0.52 | 0.71 | 1.67 |
| GraDe-IF | 0.52 | 0.76 | 1.41 |
| Bridge-IF | 0.58 | 0.81 | 1.47 |
| **Hier-IF (Ours)** | **0.59** | **0.82** | **1.36** |

## 5 CONCLUSION AND DISCUSSION

In this work, we introduce Hier-IF, a novel approach to inverse folding that explicitly accounts for the hierarchical nature of protein structures. Our method refines the traditional structure-to-sequence paradigm by incorporating a multi-stage generation process, where local structures are first generated and then assembled into a full sequence. This hierarchical design leverages biological priors and improves sequence quality, addressing a significant gap in existing inverse folding techniques. While Hier-IF represents a significant advancement in the field of inverse folding, there remain several opportunities for future improvement and exploration. One potential avenue is the refinement of the local structure prediction stage, where further optimization could lead to even higher accuracy in generating the correct local motifs. Additionally, exploring the integration of more advanced generative models or incorporating evolutionary information could help enhance the robustness of the model, particularly for more complex or less-characterized proteins.

## REFERENCES


Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

Francis H Crick. On protein synthesis. In *Symp Soc Exp Biol*, volume 12, pp. 8, 1958.

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

Frédéric A Dreyer, Daniel Cutting, Constantin Schneider, Henry Kenlay, and Charlotte M Deane. Inverse folding for antibody sequence design using deep learning. *arXiv preprint arXiv:2310.19513*, 2023.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021.

Noelia Ferruz, Steffen Schmidt, and Birte Höcker. Protgpt2 is a deep unsupervised language model for protein design. *Nature communications*, 13(1):4348, 2022.

Alan R Fersht, Andreas Matouschek, and Luis Serrano. The folding of an enzyme: I. theory of protein engineering analysis of stability and pathway of protein folding. *Journal of molecular biology*, 224(3):771–782, 1992.

Zhangyang Gao, Cheng Tan, Pablo Chacón, and Stan Z Li. Pifold: Toward effective and efficient protein inverse folding. *arXiv preprint arXiv:2209.12643*, 2022a.

Zhangyang Gao, Cheng Tan, and Stan Z Li. Alphadesign: A graph protein design method and benchmark on alphafolddb. *arXiv preprint arXiv:2202.01079*, 2022b.

Zhangyang Gao, Cheng Tan, and Stan Z Li. Knowledge-design: Pushing the limit of protein design via knowledge refinement. *arXiv preprint arXiv:2305.15151*, 2023a.

Zhangyang Gao, Cheng Tan, Yijie Zhang, Xingran Chen, Lirong Wu, and Stan Z Li. Proteininvbench: Benchmarking protein inverse folding on diverse tasks, models, and metrics. *Advances in Neural Information Processing Systems*, 36:68207–68220, 2023b.

Zhangyang Gao, Jue Wang, Cheng Tan, Lirong Wu, Yufei Huang, Siyuan Li, Zhirui Ye, and Stan Z Li. Uniif: Unified molecule inverse folding. *Advances in Neural Information Processing Systems*, 37:135843–135860, 2024.

Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, et al. Proteina: Scaling flow-based protein structure generative models. *arXiv preprint arXiv:2503.00710*, 2025.

Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *Science*, pp. eads0018, 2025.

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

Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. In *International conference on machine learning*, pp. 8946–8970. PMLR, 2022.

John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. *Advances in neural information processing systems*, 32, 2019.

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael JL Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*, 2020.

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

Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

Nobuyasu Koga, Rie Tatsumi-Koga, Gaohua Liu, Rong Xiao, Thomas B Acton, Gaetano T Montelione, and David Baker. Principles for designing ideal protein structures. *Nature*, 491(7423): 222–227, 2012.

Brian Kuhlman and Philip Bradley. Advances in protein structure prediction and design. *Nature reviews molecular cell biology*, 20(11):681–697, 2019.

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

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution. 2023.

Jiarui Lu, Xiaoyin Chen, Stephen Zhewen Lu, Chence Shi, Hongyu Guo, Yoshua Bengio, and Jian Tang. Structure language models for protein conformation generation. *arXiv preprint arXiv:2410.18403*, 2024.

Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41(8): 1099–1106, 2023.

Marc A Martí-Renom, Ashley C Stuart, András Fiser, Roberto Sánchez, Francisco Melo, and Andrej Šali. Comparative protein structure modeling of genes and genomes. *Annual review of biophysics and biomolecular structure*, 29(1):291–325, 2000.

Christine A Orengo, Alex D Michie, Susan Jones, David T Jones, Mark B Swindells, and Janet M Thornton. Cath–a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. In *International Conference on Machine Learning*, pp. 8844–8856. PMLR, 2021.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Arthur G Street and Stephen L Mayo. Computational protein design. *Structure*, 7(5):R105–R109, 1999.

Jin Su, Chenchen Han, Yuyang Zhou, Junjie Shan, Xibin Zhou, and Fajie Yuan. Saprot: Protein language modeling with structure-aware vocabulary. *bioRxiv*, pp. 2023–10, 2023.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Michel van Kempen, Stephanie S Kim, Charlotte Tumescheit, Milot Mirdita, Cameron LM Gilchrist, Johannes Söding, and Martin Steinegger. Foldseek: fast and accurate protein structure search. *Biorxiv*, pp. 2022–02, 2022.

Michel Van Kempen, Stephanie S Kim, Charlotte Tumescheit, Milot Mirdita, Jeongjae Lee, Cameron LM Gilchrist, Johannes Söding, and Martin Steinegger. Fast and accurate protein structure search with foldseek. *Nature biotechnology*, 42(2):243–246, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. *arXiv preprint arXiv:2402.18567*, 2024a.

Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Dplm-2: A multimodal diffusion protein language model. *arXiv preprint arXiv:2410.13782*, 2024b.

James C Whisstock and Arthur M Lesk. Prediction of protein function from protein sequence and structure. *Quarterly reviews of biophysics*, 36(3):307–340, 2003.

Fei Ye, Zaixiang Zheng, Dongyu Xue, Yuning Shen, Lihao Wang, Yiming Ma, Yan Wang, Xinyou Wang, Xiangxin Zhou, and Quanquan Gu. Proteinbench: A holistic evaluation of protein foundation models. *arXiv preprint arXiv:2409.06744*, 2024.

Kai Yi, Bingxin Zhou, Yiqing Shen, Pietro Liò, and Yuguang Wang. Graph denoising diffusion for inverse protein folding. *Advances in Neural Information Processing Systems*, 36:10238–10257, 2023.

Jason Yim, Andrew Campbell, Emile Mathieu, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Frank Noé, et al. Improved motif-scaffolding with se (3) flow matching. *ArXiv*, pp. arXiv–2401, 2024.

Zaixiang Zheng, Yifan Deng, Dongyu Xue, Yi Zhou, Fei Ye, and Quanquan Gu. Structure-informed language models are protein designers. In *International conference on machine learning*, pp. 42317–42338. PMLR, 2023.

Yiheng Zhu, Jialu Wu, Qiuyi Li, Jiahuan Yan, Mingze Yin, Wei Wu, Mingyang Li, Jieping Ye, Zheng Wang, and Jian Wu. Bridge-if: Learning inverse protein folding with markov bridges. *arXiv preprint arXiv:2411.02120*, 2024.

# A IMPLEMENTATION DETAILS

## A.1 BASELINE EVALUATION

In our experiments, we compare Hier-IF with several approaches, including GraphTrans (Ingraham et al., 2019), GVP (Jing et al., 2020), PiFold (Gao et al., 2022a), AlphaDesign (Gao et al., 2022b), ProteinMPNN (Dauparas et al., 2022), ESM-IF (Hsu et al., 2022), LM-Design (Zheng et al., 2023) and KW-Design (Gao et al., 2023a). We will describe the implementation as follow:

- **GraphTrans**: They introduce conditional language models for protein sequence that directly condition on a graph specification of the target structure. They can efficiently captures the complex dependencies in proteins by focusing on those that are long-range in sequence but local in 3D space.
- **GVP**: They extend standard dense layers to operate on collections of Euclidean vectors. Utilizing such layers can help model to leverage the geometric and relational aspects.
- **PiFold**: They propose a novel residue featurizer to learn multi-scale residue interactions and PiFold can generate the sequence in one shot as a result it can achieve a great effectiveness.
- **AlphaDesign**:They use AlphaFoldDB to establish a new graph-based benchmark. Furthermore, they use a simplified graph transformer encoder to introduce protein angles and propose a confidence-aware protein decoder. Such modules can efficiently improve the performance.
- **ProteinMPNN**: They begins from a message passing neural network which can effectively explore the information of backbones. Then, they replaced the fixed decoding order with an order agnostic autoregressive model, which means the decoding order is random. Such strategy can efficiently improve the performance for inverse folding.
- **ESM-IF**:They augments the inverse folding dataset with folding data created by AlphaFold2. Training with additional data, the performance achieves a great progress.
- **LM-Design**: They conduct a lightweight structural adapter which is implanted into protein language models and endows it with structural awareness.
- **KW-Design**: They propose a knowledge-aware module that refines low-quality residues. Furthermore, they introduce a memory-retrieval mechanism to save more training time.

## A.2 DATASETS

In our experiments, we evaluate the proposed Hier-IF model on two widely used protein domain datasets: CATH4.2 (Orengo et al., 1997) and CATH4.3 (Hsu et al., 2022). These datasets provide a reliable benchmark for assessing the capacity of models to generalize across diverse protein topologies. For CATH4.2, we follow the experimental setup described in GraphTrans (Ingraham et al., 2019). We utilize all protein domains from the CATH4.2 40% non-redundant set. From these domains, we extract the corresponding full protein chains, filtering out those longer than 500 residues to control computational complexity and ensure consistency with prior work. The resulting dataset is then split based on CATH topology classification codes into training, validation, and test sets, maintaining an 80/10/10 ratio. This ensures that each topology appears in only one of the three sets, promoting rigorous generalization evaluation. For CATH4.3, we adopt the protocol introduced in ESM-IF (Hsu et al., 2022). Specifically, we split the topology classification codes from CATH4.3 into training, validation, and test sets using the same 80/10/10 ratio. We extract full chains of no more than 500 residues, ensuring consistency across datasets and alignment with computational constraints.

## A.3 TRAINING

The models are trained up to 100 epochs by default using the Adam optimizer on NVIDIA A6000s. We use the training setting as ProteinMPNN, where the batch size is set to 6000 residues. Furthermore, we employ the Noam learning rate scheduler (Vaswani et al., 2017), which is widely used in transformer-based models. This scheduler starts with a warm-up phase where the learning rate increases linearly, followed by a decay phase where it decreases proportionally to the inverse square root of the training step.

# B Details of Hier-IF

## B.1 Masked Diffusion Model

We formulate the mask-based diffusion process over discrete or structured protein representations (e.g., residue identities or torsion angles) using a forward process that progressively masks input tokens, and a reverse process that learns to recover them.

### B.1.1 Forward Process (Masking)

Given a protein representation $x_0 \in \mathcal{X}^L$, where $L$ is the sequence length, the forward process defines a sequence of increasingly masked inputs $\{x_t\}_{t=0}^T$. At each timestep $t$, a binary mask $m_t \in \{0,1\}^L$ is sampled from a masking schedule $q(m_t \mid t)$, and we define:

$$x_t = m_t \odot \texttt{[MASK]} + (1 - m_t) \odot x_{t-1}$$

where $\odot$ denotes elementwise multiplication, and `[MASK]` is a special token representing an unknown residue or feature.

The marginal distribution of $x_t$ given $x_0$ can be written as:

$$q(x_t \mid x_0) = \mathbb{E}_{m_t}[x_t \mid x_0] = m_t \odot \texttt{[MASK]} + (1 - m_t) \odot x_0$$

### B.1.2 Reverse Process (Unmasking)

The denoising model $p_\theta$ learns to reconstruct the original input by predicting the masked elements:

$$p_\theta(x_{t-1} \mid x_t) = \prod_{i \in \mathcal{M}_t} p_\theta(x_{t-1}^{(i)} \mid x_t)$$

where $\mathcal{M}_t = \{i \mid m_t^{(i)} = 1\}$ is the set of positions masked at timestep $t$.

For protein sequence modeling, each term is modeled by a categorical distribution over the 20 standard amino acids:

$$p_\theta(x_{t-1}^{(i)} = a \mid x_t) = \text{softmax}(f_\theta(x_t))_a \tag{7}$$

where $f_\theta$ is a neural network (e.g., a transformer) that outputs logits for each residue position.

## B.2 Mask Generator

To enable a structure-aware and learnable masking strategy, we introduce a *mask generator* that produces a soft masking score for each residue at every diffusion timestep. The mask generator is implemented as a 3-layer multi-layer perceptron (MLP), which takes as input both the DSSP-derived secondary structure and the current diffusion timestep.

Each residue's secondary structure $s_i$ is encoded as a one-hot vector:

$$\text{onehot}(s_i) = \begin{cases} [1,0,0], & \text{if } s_i = \text{H (helix)} \\ [0,1,0], & \text{if } s_i = \text{E (strand)} \\ [0,0,1], & \text{if } s_i = \text{C (coil)} \end{cases}$$

The mask generator outputs a soft mask $\tilde{m} \in (0,1)^L$, where each element $\tilde{m}_i$ represents the probability of masking the $i$-th residue. To encourage discrete masking behavior during training, a hard mask $m \in \{0,1\}^L$ is sampled from $\tilde{m}$ according to a Bernoulli distribution:

$$m_i \sim \text{Bernoulli}(\tilde{m}_i).$$

Since this sampling operation is inherently non-differentiable, it blocks gradient backpropagation through $m$. To mitigate this, we employ the Straight-Through Estimator (STE) technique, which treats the sampling step as an identity function during the backward pass. Concretely, the gradient is directly propagated from the loss with respect to the hard mask $m$ back to the soft mask $\tilde{m}$:

$$\frac{\partial \mathcal{L}}{\partial \tilde{m}} \approx \frac{\partial \mathcal{L}}{\partial m}.$$

Alternatively, training can be conducted solely with the soft mask $\tilde{m}$ to maintain full differentiability, at the cost of losing discrete mask interpretability.

This approach enables the mask generator to learn meaningful, time-dependent masking strategies that balance discrete perturbations with smooth optimization.

—

The diffusion timestep $t$ is represented as a scalar, either as an integer in the diffusion schedule $t \in \{0, \dots, T\}$ or normalized to a continuous value in the range $[0, 1]$. This scalar is then mapped to a higher-dimensional embedding via a small learnable MLP:

$$\tau(t) = W_2 \cdot \mathrm{ReLU}(W_1 t + b_1) + b_2, \quad \tau(t) \in \mathbb{R}^{d_t},$$

where $W_1 \in \mathbb{R}^{h \times 1}$, $W_2 \in \mathbb{R}^{d_t \times h}$, and $h$ denotes the hidden dimension size.

For each residue $i$, the input vector to the mask generator is formed by concatenating the one-hot encoded secondary structure and the time embedding:

$$z_i = \mathrm{concat}\big(\mathrm{onehot}(s_i),\, \tau(t)\big) \in \mathbb{R}^{3+d_t}.$$

The mask generator $g_\phi$ is a 3-layer MLP that maps $z_i$ to the soft mask score:

$$\tilde{m}_i = \sigma\left(W_3 \cdot \mathrm{ReLU}\big(W_2 \cdot \mathrm{ReLU}(W_1 z_i + b_1) + b_2\big) + b_3\right),$$

where $\sigma(\cdot)$ denotes the sigmoid activation, ensuring $\tilde{m}_i \in (0, 1)$ represents the masking probability for residue $i$.

—

The resulting soft mask $\tilde{m} \in (0, 1)^L$ is sampled during training to produce a hard mask $m$, while at inference time, a deterministic mask can be obtained by thresholding or selecting the top-$k$ residues with highest mask scores.

This design allows the model to learn residue-level, structure- and time-aware masking strategies that adapt dynamically according to both the diffusion stage and the protein secondary structure context.

### B.3 Soft vs. Hard Masking

We distinguish between two types of masking strategies: *soft* masking and *hard* masking, based on how the mask is applied during training and inference.

**Soft masking.** In soft masking, the output $\tilde{m} \in (0, 1)^L$ from the mask generator is directly interpreted as a probabilistic or fractional mask. This enables smooth, differentiable updates, and is especially useful when the mask is applied via interpolation:

$$\hat{x} = \tilde{m} \odot \text{noise} + (1 - \tilde{m}) \odot x$$

Here, each position is softly corrupted in proportion to its predicted mask score. This allows gradients to flow through the masking mechanism during backpropagation, and facilitates end-to-end training.

**Hard masking.** In hard masking, the soft mask $\tilde{m}$ is used to sample a binary mask $m \in \{0, 1\}^L$, typically as:

$$m_i \sim \text{Bernoulli}(\tilde{m}_i)$$

This sampled mask is then used to fully corrupt certain positions and leave others untouched. While this approach aligns more closely with classical denoising objectives and provides stronger perturbations, the sampling process is non-differentiable and introduces variance during training. Common workarounds include using the Straight-Through Estimator (STE) or Gumbel-softmax relaxation.

**Comparison.** Soft masking provides smoother training dynamics and supports gradient-based optimization of the masking policy. Hard masking, in contrast, introduces more discrete and interpretable corruption patterns, often preferred at inference time or when seeking sharper denoising effects.

In our implementation, we employ soft masking during training to ensure gradient flow, and optionally apply hard masking at inference via either thresholding or top-$k$ selection based on the predicted $\tilde{m}$.

### B.4 DSSP Secondary Structure Annotation

DSSP (Define Secondary Structure of Proteins) (Kabsch & Sander, 1983) is a widely used algorithm for assigning secondary structure elements to each residue in a protein based on its 3D atomic coordinates. It categorizes residues into eight classes:

$$\{\mathrm{H}, \mathrm{B}, \mathrm{E}, \mathrm{G}, \mathrm{I}, \mathrm{T}, \mathrm{S}, \mathrm{C}\}$$

where

- H: Alpha-helix
- B: Isolated beta-bridge residue
- E: Extended strand, participates in beta ladder
- G: $3_{10}$ helix
- I: Pi-helix
- T: Turn
- S: Bend
- C: Coil (none of the above)

For many applications, including ours, a simplified secondary structure representation is preferred. We map the original 8 classes into 3 broader categories, commonly referred to as *Helix (H)*, *Strand (E)*, and *Coil (C)*, as follows:

$$\begin{cases} \mathrm{H} = \{\mathrm{H}, \mathrm{G}, \mathrm{I}\} & \text{(Helices)} \\ \mathrm{E} = \{\mathrm{E}, \mathrm{B}\} & \text{(Strands)} \\ \mathrm{C} = \{\mathrm{T}, \mathrm{S}, \mathrm{C}\} & \text{(Coils/Loops)} \end{cases}$$

This reduction simplifies the secondary structure annotation while preserving the key structural motifs relevant for downstream modeling and masking tasks.

## C Evaluation Metric

**Median Recovery** Median Recovery measures the similarity between generated protein sequences and target reference sequences, commonly used to evaluate reconstruction accuracy. Given a set of sequence pairs $\{(S_i^{\mathrm{gen}}, S_i^{\mathrm{ref}})\}_{i=1}^N$, the recovery for each pair is defined as the proportion of matching amino acids at corresponding positions:

$$\mathrm{Recovery}_i = \frac{1}{L_i} \sum_{j=1}^{L_i} \mathbb{I}\big(S_{i,j}^{\mathrm{gen}} = S_{i,j}^{\mathrm{ref}}\big),$$

where $L_i$ is the length of the $i$-th sequence and $\mathbb{I}(\cdot)$ is the indicator function. Median Recovery is the median of all individual recoveries:

$$\text{Median Recovery} = \mathrm{median}\big(\{\mathrm{Recovery}_i\}_{i=1}^N\big).$$

A higher median recovery indicates better agreement between generated and reference sequences.

**Perplexity** Perplexity is a widely used metric to assess the quality of probabilistic sequence models, reflecting how well a model predicts a given protein sequence. For a sequence $S = (s_1, s_2, \ldots, s_L)$ and a model parameterized by $\theta$ that estimates conditional probabilities $p_\theta(s_j | s_{<j})$, the log-likelihood is:

$$\log p_\theta(S) = \sum_{j=1}^{L} \log p_\theta(s_j | s_{<j}).$$

The perplexity of the sequence is defined as:

$$\mathrm{Perplexity}(S) = \exp\left(-\frac{1}{L} \log p_\theta(S)\right).$$

Lower perplexity indicates that the model predicts the sequence with higher confidence and better accuracy.

**Designability** A protein backbone is considered *designable* if there exists at least one amino acid sequence that folds into that structure. Our evaluation of designability follows the methodology outlined by Yim et al. (2024). For each backbone generated by a model, we produce eight candidate sequences using ProteinMPNN (Dauparas et al., 2022) with a sampling temperature of 0.1. Each designed sequence is then folded using ESMFold (Lin et al., 2023) to predict its 3D structure.

We calculate the root mean square deviation (RMSD) between each predicted structure and the model's original backbone. The lowest RMSD among the eight predictions, termed the self-consistency RMSD (scRMSD), is used to evaluate designability. A backbone is classified as designable if its scRMSD is below a threshold of 2 Å.

The overall designability score for a model is computed as the fraction of generated backbones that meet this criterion:

$$\text{Designability} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\big(\text{scRMSD}(S_i) < 2\,\text{Å}\big),$$

where $N$ is the total number of generated backbones and $\mathbf{1}(\cdot)$ is the indicator function.

This approach quantitatively measures the practical feasibility of designing sequences that reliably fold back to the intended structures, thus reflecting the quality of the generative model.

**Structural Diversity via Foldseek** Structural diversity measures the extent to which generated protein sequences fold into a variety of distinct structural conformations, which is important for exploring novel folds and functions.

Foldseek (Van Kempen et al., 2024) is a state-of-the-art tool for fast and sensitive structural comparison and clustering of large protein datasets. After predicting 3D structures for designed sequences, Foldseek is used to perform pairwise structural alignments and group proteins into clusters based on their similarity.

Diversity is then quantified by metrics such as:

- The number of distinct structural clusters detected at a given similarity cutoff, reflecting the breadth of structural exploration.
- Average pairwise structural dissimilarity (e.g., average RMSD or 1 - TM-score) among the generated structures.

## D Different Sampling strategy

| Method | Recovery | | | Perplexity | | |
|---|---|---|---|---|---|---|
| | Short | Single-Chain | All | Short | Single-Chain | All |
| Iterative Refinement | 0.40 | 0.41 | 0.53 | 6.97 | 6.02 | 5.80 |
| Time-aware Mask Generator | **0.46** | **0.46** | **0.59** | **5.46** | **5.17** | **4.01** |

Masked diffusion models generate discrete sequences by iteratively predicting masked tokens. While these models are efficient and parallelizable, initial predictions may suffer from inconsistencies or errors, particularly when conditioned on complex structures (e.g., protein 3D conformation). To address this, **iterative refinement** is introduced as a mechanism to improve generation quality by progressively revising uncertain predictions across multiple steps.

Let a discrete sequence of length $L$ be denoted by $\mathbf{x} = [x_1, x_2, \ldots, x_L]$, and let $\mathbf{x}^{(0)}$ be the initial input where some tokens are masked. The goal is to iteratively update $\mathbf{x}^{(t)}$ to produce a coherent sequence $\hat{\mathbf{x}}^{(T)}$ after $T$ refinement steps.

At each iteration $t = 1, \ldots, T$, the refinement process involves three stages:

1. **Prediction:** The masked model $f_\theta$ predicts the token distribution given the current sequence:

$$\hat{\mathbf{x}}^{(t)} = f_\theta(\mathbf{x}^{(t-1)}, \text{cond})$$

where `cond` denotes optional conditioning information (e.g., structural constraints).

2. **Remasking:** Based on prediction confidence (e.g., entropy or max softmax probability), a subset $\mathcal{M}^{(t)}$ of uncertain positions is selected to be masked again:

$$\mathcal{M}^{(t)} = \text{TopKUncertain}(\hat{\mathbf{x}}^{(t)})$$

3. **Update:** The input for the next step is updated as:

$$x_i^{(t)} = \begin{cases} \hat{x}_i^{(t)}, & \text{if } i \notin \mathcal{M}^{(t)} \\ \texttt{[MASK]}, & \text{if } i \in \mathcal{M}^{(t)} \end{cases}$$

This process is repeated for $T$ iterations to progressively refine the prediction toward a coherent and high-quality output.

Following Zheng et al. (2023), we implement the iterative refinement strategy for sequences generation. Compared with the time-aware Mask Generator proposed, we find our methods can achieve better performance. Because of the awareness of the hierarchical information, Hier-IF can design the protein sequence with higher recovery and perplexity.

# E Additional Results

## E.1 Generalization on Independent Test Sets (TS50 and TS500)

To rigorously evaluate the generalization capability of Hier-IF on unseen protein structures, we employed two widely recognized independent test sets: TS50 and TS500.

- TS50: This dataset consists of 50 diverse protein structures. It was originally curated to measure the performance of protein design models on structures that are strictly strictly separated from the training data, ensuring no sequence or structural redundancy.
- TS500: This is a larger and more challenging dataset containing 500 protein structures. It covers a broader range of protein folds and topologies, providing a more comprehensive assessment of a model's robustness and ability to handle complex structural variations.

The detailed performance comparison on these datasets is presented in Table 5. Hier-IF consistently outperforms the baselines, particularly on the larger TS500 set, demonstrating its strong generalization ability.

Table 5: **Generalization performance on independent test sets.** We compare Hier-IF with state-of-the-art methods on TS50 and TS500 datasets. The best results are highlighted in **bold**.

| Dataset | Model | Recovery (↑) | Perplexity (↓) |
|---|---|---|---|
| TS50 | ProteinMPNN | 54.43 | 3.93 |
| | PiFold | 58.72 | 3.86 |
| | LM-Design | 57.89 | 3.50 |
| | **Hier-IF (Ours)** | **58.01** | **3.47** |
| TS500 | ProteinMPNN | 58.08 | 3.53 |
| | PiFold | 60.42 | 3.44 |
| | LM-Design | 64.30 | 3.19 |
| | **Hier-IF (Ours)** | **66.42** | **2.83** |

## E.2 Details of In Silico Folding Validation

In Section 4.4, we reported the structural consistency results. Here, we provide the detailed setup for this validation.

We utilized openfold to predict the 3D structures of the sequences generated by Hier-IF and baseline models. The structural quality was evaluated using two key metrics:

- **TM-score**: A metric for assessing the topological similarity of protein structures. It ranges from 0 to 1, with higher scores indicating better structural overlap. A score above 0.5 typically implies roughly the same fold.
- **scRMSD** (side-chain Root Mean Square Deviation): This metric measures the deviation of side-chain packing between the predicted structure and the native backbone. Lower scRMSD values indicate higher precision in side-chain packing and overall structural consistency.

As shown in the main text, Hier-IF achieves superior TM-score and competitive scRMSD, validating that the hierarchical information effectively guides the model to generate foldable sequences.

### E.3 Ablation Study

In Hier-IF training, we introduce a bidireactional reconstruction loss. Furthermore, we introduce a sampling strategy with structure-aware remask mechanism. This section evalution the effects of these technic on CATH4.2 dataset. The Table 6 shows the results. Please refer Appendix for more details.

Table 6: Ablation results on different settings

| Bidireactional reconstruction loss | Structure-aware remask | Recovery | | | Perplexity | | |
|---|---|---|---|---|---|---|---|
| | | Short | Single-chain | All | Short | Single-chain | All |
| ✗ | ✗ | 0.28 | 0.29 | 0.36 | 8.27 | 8.73 | 6.61 |
| ✓ | ✗ | 0.34 | 0.38 | 0.51 | 7.23 | 6.42 | 4.82 |
| ✗ | ✓ | 0.42 | 0.43 | 0.53 | 6.02 | 5.90 | 4.36 |
| ✓ | ✓ | 0.46 | 0.46 | 0.59 | 5.46 | 5.17 | 4.01 |

To demonstrate the necessity of our proposed iterative generation and remasking mechanism, we analyzed how the quality of generated sequences evolves throughout the generation process. Specifically, we compared our structure-aware strategy (DSSP-based) against a random remasking baseline.

Table 7: Comparison of sequence recovery during the iterative generation process (50 steps).

| **Method** | **Step=0** | **Step=10** | **Step=20** | **Step=30** | **Step=40** | **Step=50** |
|---|---|---|---|---|---|---|
| Random | 0.00 | 0.23 | 0.32 | 0.42 | 0.48 | 0.51 |
| **DSSP-based (Ours)** | **0.00** | **0.24** | **0.36** | **0.45** | **0.52** | **0.58** |

Table 7 presents the sequence recovery rates recorded at different intervals of the 50-step sampling process. As shown in the results, the structure-aware strategy consistently outperforms the random baseline at every interval of the generation trajectory. This comparison confirms the effectiveness of our proposed mechanism, as it utilizes structural cues to guide the model toward higher-quality sequences more efficiently than random masking.