TSINGHUA SCIENCE AND TECHNOLOGY
DOI: 10.26599/TST.2026.9010082

Research/Review Article

MixInfoFold: A Method For Iteratively Generating Better Sequences

An Zeng, CanHui Chen, JinRong Li, BaoYao Yang, Dan Pan✉

Abstract: Current AlphaFold3-based protein structure prediction has reached remarkable levels, but sequences that can be applied in practice are mostly generated using physics-based methods. Here, we introduce a graph representation-based protein sequence generation method: MixInfoFold which incorporates new noise addition and feature extraction methods, as well as our designed encoder-decoder called MixInfo. The noise addition method adds Gaussian noise positively correlated with the sequence length at the end of the main-chain, which enhances the model's ability to understand the dynamic changes in protein structure. The feature extraction method includes a new feature calculated by assessing the area and distance of the main-chain atoms. The MixInfo iteratively computes the relationships between edge and node features, enabling a deeper exploration of the implicit sequence information within protein structures. Compared to the best model, our model achieves an improvement of 0.80%, 0.89%, and 3.28% in sequence recovery rates on the CATH4.2, CATH4.3, and Model$_{final}$ datasets, respectively. Additionally, our model generates sequences faster than other methods, and generates sequences closely resemble natural proteins, indicating the model's feasibility and potential value in practical applications.

Key words: deep learning; protein design; iterative calculation; generating models

1 Introduction

Proteins are the primary bearers of biological activities, playing various crucial roles such as maintaining normal metabolism and repairing tissues. In biotechnological research, generating protein sequences that can form a given three dimensional structure is a highly important subject, known as protein generation for fixed structures. The goal of protein generation is to find an amino acid sequence that best fits the target structure. This process is both highly complex and extremely sensitive to minor variations,the challenge in protein generation lies is that even minor structural differences can correspond to significantly different sequences, and incorrect sequences cannot fold into the expected target structure[1], resulting in failure of the entire task. The conventional methods for protein sequence generation mainly rely on optimizing side-chain types and conformations based on empirical energy functions[2, 3] which are derived either from summarizing physical laws (such as RosettaDesign[4], Proteus[5], and EvoEF2[6]) or from statistical analysis of data (such as ABACUS[7] and TERM[8]). The empirical energy functions often approximate the complex multi-body molecular interactions as linear combinations of monomer and dimer terms when describing them[9]. Despite continuous optimization and improvement of traditional energy functions, this approximation remains a fundamental limiting factor in their accuracy.

Deep learning is an efficient and flexible approach that breaks free from the limitations of approximation methods. Currently, many researchers have successfully applied deep learning in the field of protein generation[10], including enzymes[11] and macromolecular complexes[12]. Among them, deep learning graph models exhibit remarkable flexibility and efficient computational capability in representing proteins[13–15], thus leading to an increasingly widespread application of such models in the field of protein generation. For example, by constructing a graph neural network using residue relationship features and leveraging Transformer[16] to capture these relationships[17], the sequence recovery rate can be improved to 40%, a result that significantly surpasses traditional energy-based methods (Rosetta's sequence recovery rate is 33.1% (Table 4)). The distance features between main-

● An Zeng ,CanHui Chen,Jinrong Li,and Baoyao Yang are with the Faculties of Computer, GuangDong University of Technology, Guangzhou and 510006, China. E-mail: zengan@gdut.edu.cn;2112405122@mail2.gdut.edu.cn; 2112205086@mail2.gdut.edu.cn; byyang@comp.hkbu.edu.hk.
● Dan Pan is with the Faculty of Electronics and Information Technology, Guangdong Polytechnic Normal University, Guangzhou and 510665, China. E-mail: pandan@gpnu.edu.cn.
Manuscript received: 2026-04-30; revised: 2026-04-30; accepted: 2026-08-02

chain atoms proposed by Dauparas[3] increased the sequence recovery rate from 40% to 49%. However, current deep learning graph models mostly rely on geometric features such as dihedral angles and do not delve into the important features that exist among main-chain atoms[18]. While there have been studies[3,17] proposing strategies for handling three-dimensional structure of varying lengths, such as the full-zero noise and on-chain noise methods, the former overlooks the dynamic changes in protein structures and the latter directly changes the main-chain structure. As a result, many deep learning graph models still cannot achieve a sequence recovery rate of more than 50% on the CATH dataset[18].

Here, we propose a novel protein design method called MixInfoFold. It first adds Gaussian noise that matches the sequence length to the end of the main-chain structure using a noise addition method that we designed. This noise addition method helps the model ignore insignificant subtle changes and increases the diversity of the data. Experiments have shown (Table 6) that this method can improve model performance and outperforms other methods (with an accuracy improvement of approximately 0.55%). Following this, it extracts features such as angles, relative positions, spatial locations, and sizes from the main chain structure with added noise, iteratively computing these features through our designed encoder and decoder to generate better sequences. For each residue, we construct a new feature: the area formed by the N, C, and O atoms on the main-chain and the distance from the C_α atom to this plane. We consider that the N atom on the main-chain is connected to the C_α atom by a single bond, making it more susceptible to relative displacement influenced by side chain groups. Feature ablation experiments (Table 7) indicate that this feature contributes approximately 1.2% to the sequence recovery rate. We designed node feature update structures and edge feature update structures for the encoder-decoder. Through these two update structures, we iteratively compute the attention of the input edge features and node features, completing the feature updates. In the decoder, to enhance the model's ability to learn the relationships between sequences and structures, we integrate the true sequences into the edge features and adopt a sequential decoding method (refer Section 2.4.1) to process.

We highlight our significant contributions and research achievements as follows:

(1).We propose a model named MixInfoFold for generating protein sequences based on three-dimensional structure.It performs multiple calculations of attention between edge and node features to update features.This model effectively captures and learns the intrinsic patterns and characteristics of sequences from structural information, leading to higher accuracy in sequence generation.Most existing models primarily use residue sequence features as the node attributes of the graph, and often neglect the construction of edge features, converting simple Cα distance relationships into one-hot encodings as edge attributes.

(2).We further propose a residue-based feature extraction method that determines amino acid types by analyzing the spatial area and interatomic distances formed by the main-chain atoms of residues.Unlike conventional approaches that rely solely on sequence-derived descriptors, our method directly leverages geometric and spatial cues from the protein backbone, enabling a more fine-grained characterization of residue properties. This geometric-aware representation significantly enriches the feature space, thereby enhancing our model's capacity to capture structure–sequence relationships, and contributes to a 1.20% performance improvement on the Model$_{final}$ dataset.

(3).Our research reveals that introducing Gaussian noise positively correlated with sequence length at the terminus of the protein main-chain can effectively improve the model's sequence recovery rate. In contrast to strategies, such as uniformly perturbing the entire main-chain or applying zero-valued noise at its terminus, our targeted noise-injection scheme selectively perturbs structural termini in a length-adaptive manner.This not only mitigates overfitting to terminal patterns but also encourages the model to learn robust sequence–structure correspondences, achieving performance in sequence recovery.

(4).The experiments demonstrate that our model outperforms the best model, with a sequence recovery rate improvement of 0.80%, 0.89%, and 3.28% on the datasets CATH4.2, CATH4.3, and Model$_{final}$, respectively.

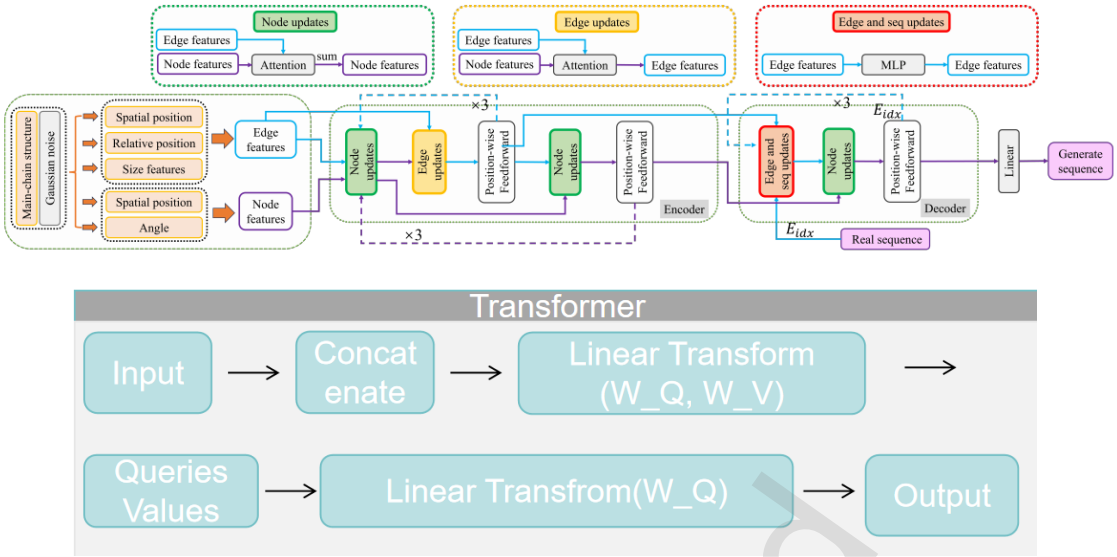

Fig. 1 The model for generating protein sequences given a three-dimensional structure. We set spatial position features, relative position features, and size features as edge features, and spatial position features and angle features as node features. After passing through three encoder layers and three decoder layers, the model outputs the generated protein sequence. Attention aggregates neighboring information, and MLPs transform features.

2 Materials and methods

2.1 Datasets

To evaluate and compare the performance of our model, we conducted experiments on three datasets:

CATH4.2 collected by Ingraham[17] (Train: 17,224 proteins, Valid: 573 proteins, Test: 1,072 proteins), CATH4.3 collected by Hsu[34] (Train: 20,160 proteins, Valid: 448 proteins, Test: 1,793 proteins), and the non-redundant dataset Model$_{final}$ from the Protein Data Bank (PDB) collected by Liu[35] (Train: 22,189 proteins, Valid: 1,433 proteins, Test: 1,433 proteins). Additionally, we compared our model with the Rosetta method[36] based on an energy function using a small-scale test set TS50 collected by Li[19]. Besides evaluating the performance of the model in generating sequences for a given structure, we used the CASP14 test set to verify the effectiveness of our proposed size feature.The CASP14 test set consists of experimentally determined but undisclosed structures,which provides a rigorous benchmark for fair evaluation.

2.2 Model introduction

In Figure 1, we show our designed model, which takes the three-dimensional structure of a protein as input and outputs the corresponding protein sequence. We learn the implicit sequence information in the structure through an **encoder-decoder** called MixInfo. The encoder captures higher-order relationships between features by iteratively computing attentionon edge features and node features. The decoder combines edge features with the actual sequence, while the node features gradually learn sequence information from the edge features, ultimately outputting the model's generated sequence. In the following sections, we will introduce the extracted features from the structure, describe the composition of edge features and node features, and present the specific architecture of the model.

2.3 Modeling of three-dimensional structure

Based on the distances between C_α atoms in the main-chain, we define the set of nearest neighboring residues forming a three-dimensional local environment around the central residue as the k-nearest. We considered the 48 neighboring residues closest to the central residue (the reason for choosing 48 is evident in Table 5) and extracted the following features for the residues. The edge features consist of the spatial position feature (V_{emb}), the relative position feature (P_{emb}) and the size feature (R_{size}). The node features consist of the angle feature (V_{angl}) and the spatial position feature (V_{emb}). (refer Figure 2)

2.3.1 Angle feature (V_{angl})

We computed the dihedral angles $(\alpha,\ \beta,\ \gamma)$ and

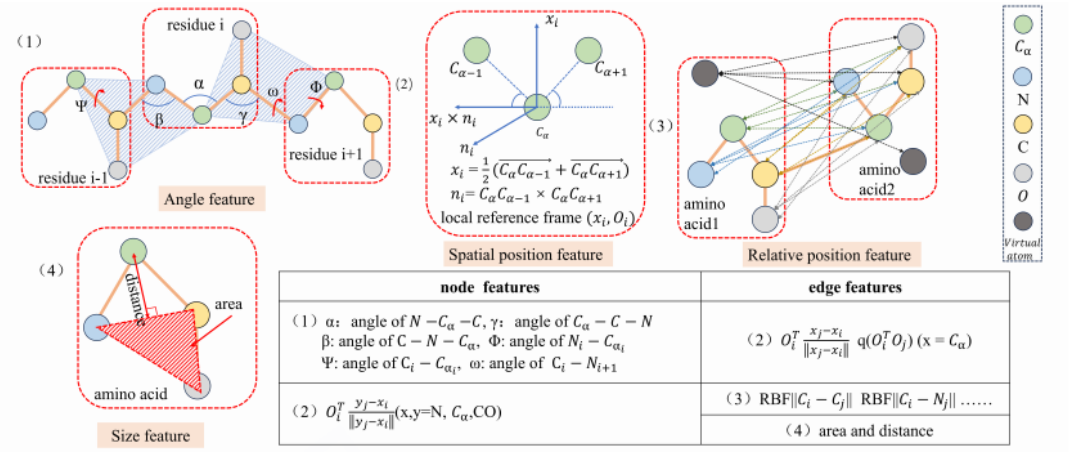

Fig. 2 Feature representation diagram. The diagram illustrates four types of residue features constructed based on the three-dimensional structure of the protein, with the annotation numbers corresponding to the features described above. O_i represents the local coordinate system (x_i, O_i) with x_i as the local direction vector of the C_α atom, n_i as the normal vector, and $x_i \times n_i$ as the third basis vector.

torsion angles (ϕ, ψ, ω) of residues and used these angles as node features. These angles are crucial for reconstructing the conformation of the protein backbone by the model. The specific representations of these angles are shown in Figure 2.

2.3.2 Spatial position feature (V_{emb})

We define a local coordinate system:

$$O_i = (x_i, n_i, x_i \times n_i), \tag{1}$$

where x_i is the angle bisector of the angle formed by $C_{\alpha-1}-C_\alpha$ and $C_{\alpha+1}-C_\alpha$, n_i is the unit normal vector to the plane of $C_{\alpha-1}C_\alpha C_{\alpha+1}$. We describe spatial positional features (V_{emb}) by rigid-body transformations between reference frames (x_i, O_i) and (x_j, O_j):

$$V_{emb} = (O_i^T \frac{x_j - x_i}{\|x_j - x_i\|}, q(O_i^T O_j)). \tag{2}$$

The spatial position feature consists of two parts. The first part involves transforming the coordinate difference vectors of main-chain nodes relative to the central residue from the global coordinate system to a reference coordinate system (x_i, O_i), and then obtaining directional information within this reference system. The second part is the spatial rotation matrix $(O_i^T O_j)$ obtained by multiplying the reference system (x_i, O_i) with the reference system of the k-nearest. Here, q represents the quaternion method, used to represent the direction of rotation and the magnitude of principal stress in the x, y, and z directions of the spatial rotation matrix $(O_i^T O_j)$[32]. Since the spatial rotation matrix $(O_i^T O_j)$ contains neighboring residues, the node feature contains only the first part of this feature.

2.3.3 Relative position feature (P_{emb})

We first design four virtual atoms based on the absolute positions of the main-chain atoms: $V_1, \ldots, V_4$. The calculation formula for the virtual atoms is as follows:

$$\begin{aligned} V_i = {} & a(C_\alpha - N) + b(C - C_\alpha) \\ & + c\,((C_\alpha - N) \times (C - C_\alpha)) + C_\alpha \end{aligned} \tag{3}$$

where a, b, and c are randomly set constants with absolute values less than one. We calculated the relative distances between each pair of atoms (four main-chain atoms and four virtual atoms) and combined the calculated relative distances into edge features using k-nearest. Finally, we employ radial basis functions (RBF) to transform and process the calculated relative distances, enabling more effective utilization of this distance information in subsequent analyses.

$$P_{NN} = RBF^*(\|N_i - N_j\|)_{j \in N(i,k)}, \tag{4}$$

$$P_{emb} = P_{NN} + P_{nC_\alpha} + \ldots + P_{V_4 V_4}. \tag{5}$$

2.3.4 Size feature (R_{size})

We believe that the relative positions between side-chain atoms and main-chain atoms can have an impact. Specifically, the N atom on the main-chain is connected to the C_α atom via a single bond, making it more susceptible to significant relative displacements due to perturbations from side-chain groups. Thus, we propose a new feature: by calculating the area of the plane formed by the three main-chain atoms (N, C, O) and the distance from

*We used 16 Gaussian RBFs isotropically spaced from 0 to 20 Angstroms.

C_α to this plane. We also use k-nearest and radial basis function (RBF) to process this feature.

$$R_{size} = RBF(area_{ij}, distance_{ij})_{j\in N(i,k)}, \quad (6)$$

$$area_i = \frac{1}{2}(X_N - X_C) \times (X_N - X_O), \quad (7)$$

$$distance_i = \frac{((X_N - X_C)\times(X_N - X_O))(X_{C_\alpha} - X_N)}{\|(X_N - X_C)\times(X_N - X_O)\|}. \quad (8)$$

In subsequent experiments, we will further verify the effectiveness of size features.

2.4 Model architecture

This section outlines the architecture of our model, which is designed to iteratively generate better protein sequences based on the three-dimensional structure. The model employs an encoder-decoder structure, named MixInfo, to compute the relationships between edge and node features. The encoder captures higher-order relationships between features by iteratively computing attention on edge features and node features. The decoder integrates the true sequences into the edge features and adopts a sequential decoding method to process the sequence information. We will describe the specific architecture of the model, including the sequential decoding method, the encoder, and the decoder.

In the MixInFo, the node features are 128-dimensional vectors representing per-residue descriptors, while the edge features are 256-dimensional vectors encoding geometric and physicochemical relationships between residues. For each residue node, we retain the 48 closest neighbors when constructing the protein graph, ensuring a balance between computational efficiency and structural coverage. A dropout rate of 0.15 is applied throughout the network to prevent overfitting and enhance generalization.

2.4.1 Sequential decoding method

Our sequence generation refers the sequential decoding approach proposed by Ingraham[17] :

$$p(s|x) = \prod_i p(s_i|x, s_{<i}). \quad (9)$$

That is, the conditional probability $p(s_i|x, s_{<i})$ of amino acid s_i at position i is influenced by two factors: the structural features of the decoder input and the actual amino acid sequence $s_{<i} = \{s_1, \ldots s_{i-1}\}$ preceding position i. The encoder updates features based solely on structural edge features $E(x)$ and node features $V(x)$, while the decoder, besides being able to utilize features from the encoder output, also incorporates sequence information from previous positions to obtain the final output sequence s_i.

2.4.2 Encoder

The encoder module consists of two structures (Figure 1): (1) the node feature update structure, (2) the edge feature update structure. The node feature update structure computes multi-head attention weights for the input edge features and node features. Since the multi-head attention weights incorporate weights involving k-nearest, we need to sum along the second-to-last dimension to obtain the updated node features. Similarly, the edge feature update structure computes multi-head attention weights using the same method. However, since the edge features themselves include k-nearest, there is no need for a summation operation.

$$\begin{aligned} V_i' &= \sum \Delta h(V_i, E_{ij}), \\ E_{ij}' &= \Delta h(V_i, E_{ij}). \end{aligned} \quad (10)$$

Here E_{ij} represents edge features, V_i represent node features. When computing attention weights, instead of directly calculating Q and K as in the Transformer paper [16], we use a multi-layer perceptron (MLP) to learn the weights (w_{ij}) and value embeddings (v_{ij}) of multi-head attention. We separately compute the weights and value embeddings to enable them to express different subspaces:

$$w_{ij} = W_3(\sigma(W_2(\sigma(W_1(V_i||E_{ij}||V_i))))), \quad (11)$$

$$v_{ij} = W_6(\sigma(W_5(\sigma(W_4(E_{ij}))))), \quad (12)$$

here W_1 to W_6 are linear layers, σ is the activation function. We calculate the attention h_i between the weight (w_{ij}) and the embedding value (v_{ij}):

$$\begin{aligned} a_{ij} &= \frac{\exp(w_{ij}')}{\sum_{j\in N(i,k)} \exp(w_{ij}')}, \\ w_{ij}' &= \frac{w_{ij}}{\sqrt{d}}. \end{aligned} \quad (13)$$

The result of each attention head is calculated as a weighted sum:

$$\begin{aligned} h_i &= \sum_{j\in N(i,k)} a_{ij} v_{ij}, \\ \Delta h &= W_o\, Concat\left(h_i^{(1)}, \ldots, h_i^{(L)}\right). \end{aligned} \quad (14)$$

We utilize residual structures to update edge features and node features, combining the two structural approaches (Figure 1). Finally, the updated edge features and node features are both processed through positional feed-forward layers, and the results are taken as the output of this layer.

2.4.3 Decoder

In the decoder, we added true sequence information to the edge features, allowing the decoder to access sequence elements in a sequential decoding method $s_{<i}$. For nodes located after node i in the sequence, the features can only be updated based on structural information ($r_{ij}^{(dec)} = (E_j, E_{idx}(V_j))$), while for nodes located before node i, additional sequence information $g(s_j)$ can be utilized.

$$\begin{aligned} r_{ij}^{(dec)} = & E_j^{(dec)}, E_{idx}(V_j^{(dec)}), g(s_j), \ i > j \\ & E_j^{(enc)}, E_{idx}(V_j^{(enc)}), 0, \ i \leq j \end{aligned} \tag{15}$$

Here, $E_j^{(dec)}$ and $V_j^{(dec)}$ are the feature embeddings of node j in the current layer of the decoder, $E_j^{(enc)}$ and $V_j^{(enc)}$ are the feature embeddings of node j in the final layer of the encoder, E_{idx} is the nearest neighbor matrix, and $g(s_j)$ is the real sequence embedding of amino acid s_j at node j. This sequential decoding approach restricts the current position to only access preceding sequence information, yet it still enables an understanding of structural information following that position.

In the decoder, we first update the edge features enriched with sequence information using a Multilayer Perceptron (MLP). Then, we use the edge feature update structure to update the node features. Due to the risk of information leakage from the attention weight calculation method in the encoder, here we compute attention weights (w_{ij}) using query embeddings (q_i) and key embeddings (k_{ij}).

$$q_i = W_q(V_i), \quad k_{ij} = W_k(E_{ij}). \tag{16}$$

$$\begin{aligned} a_{ij} &= \frac{\exp(w_{ij})}{\sum_{j \in N(i,k)} \exp(w_{ij})}, \\ \mathrm{w}_{ij} &= \frac{q_i^T k_{ij}}{\sqrt{d}}. \end{aligned} \tag{17}$$

Moreover, the other structures remain consistent with the edge feature update structure in the encoder.

2.5 Adding Gaussian noise to the end of the main-chain

To handle protein sequences of varying lengths, we added Gaussian noise with a mean of 0 and a standard deviation of $0.02\,\mathrm{nm}(0.2\,\text{Å})$ at the end of the main-chain, covering 2% of the total sequence length. This parameter selection was based on the dynamic characteristics of protein structures and computational modeling requirements. Experimental observations indicate that the natural fluctuation range of terminal residues in proteins typically falls between $0.2 - 1.0\,\text{Å}$, making the $0.2\,\text{Å}$ perturbation intensity suitable for simulating real biophysical properties without excessively disrupting structural integrity. Compared to existing methods (ProteinMPNN's more conservative perturbation of $0.1\,\text{Å}$ and PiFold's stronger perturbation of $0.5\,\text{Å}$), this compromise achieves a balance between maintaining structural rationality and enhancing data diversity. Setting the noise coverage length to 2% of the total sequence length ensures effective perturbation of $1 - 10$ terminal residues for typical proteins (approximately $100-500$ residues), matching their naturally flexible regions. This design accounts for the dynamic properties of terminal regions while avoiding excessive noise that could affect core structural areas.

3 Experiments

In all experiments, we used three layers of encoders and decoders ((refer Section 2.4)). The edge features, node features, and hidden dimensions in the model are set to 128. Additionally, the model dropout rate is set to 15%. To optimize model performance, we utilized the Adam optimizer[33] with an initial learning rate of 0.001 and the ReduceLROnPlateau learning rate scheduler. This scheduler multiplies the current learning rate by 0.1 and applies the new learning rate for further training if the perplexity of the validation set does not decrease for five consecutive epochs.

3.1 Comparison with State-of-the-Art Models

To comprehensively evaluate MixInfoFold, we compared it with the current state-of-the-art protein design method PLAID[44] and other deep learning models for protein-related tasks. PLAID achieves higher sequence recovery rates (e.g., 84.51% on $\text{Model}_{\text{final}}$ under the adapted PLAID setting), whereas MixInfoFold demonstrates substantially lower perplexity (4.74 vs 12.34). PLAID can exploit sequence-only databases that are substantially larger than structure databases because it requires only sequence inputs during training. In addition, its underlying ESM/ESMFold representation is itself built on large-scale protein

Table 1 The results of the model on the CATH4.2, CATH4.3, and Model$_{final}$ test sets. For ease of quick identification, we specifically annotate the best results in bold, while the suboptimal results are marked with underlines.

Dataset	Model	Recovery/ % (↑)			Perplexity (↓)		
		Short	Single chain	All	Short	Single chain	All
CATH4.2	GraphTrans	33.62%	34.41%	38.12%	8.88	8.82	6.91
	GCA	33.45%	34.07%	41.74%	8.47	8.34	6.56
	ProteinMPNN	38.96%	39.86%	46.35%	8.81	8.73	5.97
	PiFold	39.84%	38.53%	51.66%	6.04	6.31	4.55
	GradeIF	**45.27%**	**42.77%**	52.21%	**5.49**	**6.21**	4.35
	Our model	41.20%	41.49%	**53.01%**	6.25	**6.21**	**4.33**
CATH4.3	GraphTrans	38.19%	38.17%	41.85%	7.69	7.70	5.99
	GCA	40.03%	39.65%	46.96%	7.46	7.46	5.74
	ProteinMPNN	43.17%	43.48%	50.55%	7.66	7.59	5.08
	PiFold	45.76%	45.81%	56.07%	5.40	5.39	3.92
	GradeIF	**48.59%**	**47.99%**	56.46%	**5.31**	**5.34**	3.80
	Our model	45.34%	45.35%	**57.35%**	5.43	5.41	**3.75**
Model$_{final}$	GraphTrans	33.74%	33.94%	38.47%	8.04	8.00	7.12
	GCA	36.34%	36.47%	42.40%	7.28	7.23	6.29
	ProteinMPNN	41.44%	41.23%	50.37%	6.11	6.12	4.81
	PiFold	46.89%	46.90%	54.54%	5.34	5.34	4.13
	GradeIF	**50.13%**	**50.29%**	56.14%	**4.59**	**4.59**	3.79
	Our model	48.69%	48.80%	**59.42%**	4.74	4.74	**3.56**

sequence corpora. These two sources of large-scale sequence prior strengthen its ability to recover sequences that are structurally consistent with the target. PLAID generates sequences from the joint sequence–structure latent space of ESMFold; however, perplexity is more sensitive to whether the whole sequence follows the local transitions and global statistical regularities of natural proteins. Therefore, PLAID achieves strong recovery while remaining less competitive in perplexity. By contrast, MixInfoFold is developed directly for fixed-structure sequence design: it progressively extracts and updates node and edge features from the given structure to generate the sequence that best matches the target. Therefore, under the current inverse-folding setting, MixInfoFold is better positioned to maintain not only structural compatibility but also stronger sequence-level naturalness. To assess the biological plausibility and structural stability of generated sequences, we computed the Pearson correlation between the amino acid distributions of our model's outputs and natural proteins. MixInfoFold achieves a high correlation (0.972), which is competitive with BayesFlows[46] (Pearson = 0.98) and outperforms DeepPPIs[45] (Pearson = 0.74) and PreAffinitys[47] (Pearson = 0.833). This indicates that MixInfoFold generates sequences that closely resemble natural proteins, enhancing their potential for real-world applications.

3.2 The model performance on the test set

We evaluated the performance of our model on the CATH4.2, CATH4.3, and Model$_{final}$ datasets. We simultaneously reported the results for the short-chain subset and the single-chain subset in each test set (Table 1). We compared our model's results with a series of recent models, including GraphTrans[17], GCA[37], ProteinMPNN[3], PiFold[18], and GradeIF[42]. To ensure a fair and reliable comparison, we used the same data splitting method as the original authors when testing the model mentioned above. Our model achieved the best perplexity and sequence recovery performance on all three "All" test sets. In the single-chain and short-chain test sets, our model achieved suboptimal performance on CATH4.2 and Model$_{final}$.

This performance trend may be related to the characteristics of the different subsets. For the short-chain subsets, the relatively limited sequence length provides less structural context and fewer informative representations for the model to exploit, which may reduce the benefit of more complex feature extraction and interaction mechanisms. For the single-chain subsets, the issue

Table 2 Comparison of MixInfoFold with other models in terms of sequence recovery, perplexity, and Pearson correlation.

Model (Task)	Key Metric	Performance	MixInfoFold
PLAID	Sequence Recovery (%)	84.51	**48.69**
	Perplexity	12.34	4.74
DeepPPI	Pearson Correlation	0.74	**0.972**
BayesFlow	Pearson Correlation	0.98	**0.972**
PreAffinity	Pearson Correlation	0.833	**0.972**

Table 3 Test results for TS50 and TS500. All models were retested under the same code framework, except for those marked with †, whose results were copied from their manuscripts.

Method(Type)	TS50			TS500		
	Perplexity(↑)	Recovery/% (↑)	Worst/% (↑)	Perplexity (↓)	Recovery /% (↑)	Worst/% (↑)
Rosetta (Energy-Based)		33.58%	6.31%		30.83%	1.75%
SPIN †(MLP)		30.30%			30.30%	
SPIN2 †(MLP)		33.60%			36.60%	
Wang's model †(MLP)		33.00%			36.14%	
SPROF †(CNN)		39.16%			40.25%	
ProDCoNN †(CNN)		40.69%			42.20%	
DenseCPD †(CNN)		50.71%			55.53%	
GradeIF (Diffusion)	3.71	56.32%	35.50%	**3.23**	61.22%	3.00%
StructGNN (GNN)	5.40	43.89%	26.92%	4.98	45.69%	5.00%
GraphTrans (GNN)	5.60	42.20%	26.92%	5.16	44.66%	3.00%
GVP (GNN)	4.71	44.14%	33.73%	4.20	49.14%	**9.00%**
GCA (GNN)	5.09	47.02%	28.87%	4.72	47.74%	3.00%
AlphaDesign (GNN)	5.25	48.36%	32.31%	4.93	49.23%	3.00%
ProteinMPNN (GNN)	3.93	54.43%	37.24%	3.53	58.08%	3.00%
PiFold (GNN)	3.86	58.72%	37.93%	3.44	60.42%	3.00%
Our Model (GNN)	**3.63**	**60.64%**	**40.00%**	3.45	**62.57%**	4.73%

is not necessarily sequence length, but rather the relatively simpler organizational form compared with the more heterogeneous structures included in the "All" subsets. Since MixInfoFold is designed to capture richer geometric representations and higher-order interactions between node and edge features, its advantages are more likely to become evident when the structural diversity of the data is higher. By contrast, in short-chain and single-chain subsets, the reduced representation space or structural heterogeneity may make the gains of our design less pronounced.

We also tested other methods based on MLP (SPIN[19], SPIN2[22] and Wang's model[42]), CNN (SPROF[43], ProDCoNN[25] and DenseCPD[26]) and diffusion(GradeIF[42]) using the TS50 test set (Table 3). While Rosetta's performance surpasses early methods based on MLP, there is still a noticeable gap compared to the latest approaches. It is worth noting that generating a sequence length of 127 with Rosetta requires a time of 333.87s, whereas our method only takes around 1ms. In Figure 5 and Figure 6, we present a comparison of the generation speed of our model with that of other models. Our model performs better with fewer epochs and time.

3.3 The choice of neighboring node count

To assess the influence of the number of adjacent residues on sequence recovery, we conducted experiments on the CATH4.3 dataset incorporating 20, 30, 40, 48, and 64 neighboring residues (use the main-chain atom C_α as the reference). The experimental results indicate that increasing the number of adjacent nodes enhances the model's performance to a certain extent (Experiments 1 to 4 in Table 5). However, as the number of nodes increases beyond a certain threshold, the model's performance may even decline (Experiment 5 in Table 5). We speculate that this decline could be attributed to the introduction of excessive noise or redundancy from the additional node

Table 4 The model's sequence recovery rate on the TS50 benchmark test set.

Method (Type)	Recovery /% (↑)
Rosetta (Energy-Based)	33.58%
SPIN (MLP)	30.10%
SPIN2 (MLP)	34.12%
ProDCoNN (CNN)	50.89%
DenseCPD (CNN)	40.75%
PiFold (GNN)	58.72%
GradeIF (Diffusion)	56.32%
Our Model (GNN)	**60.64%**

Table 5 The impact of different numbers of neighboring nodes on model performance. To ensure consistency among the experimental data, we removed amino acid sequences with lengths less than 65. From the table, it can be observed that the model performs most optimally when the number of neighboring nodes is 48.

	Number of neighboring nodes	Recovery / % (↑)	Perplexity (↓)
1	20	54.58%	4.05
2	30	55.55%	3.94
3	40	56.07%	3.90
4	48	**56.40%**	**3.84**
5	64	55.90%	3.91

information, affecting the model's generalization ability. Despite a slight drop in sequence recovery rate in Experiment 5, its performance is still superior to that of Experiments 1 and 2, suggesting that our model may exhibit good performance even in more complex contextual information.

3.4 Adding noise to improve model performance

We found that adding Gaussian noise positively correlated with the sequence length at the end of the main-chain can improve the performance of the model, and performs better than the methods of adding all-zero noise at the end of the main-chain by Ingraham[17] and adding noise in the main-chain by Dauparas[3]. Compared with the zero-noise method, since proteins exhibit dynamic changes within biological organisms and the terminal regions of the main chain are typically more flexible, adding noise to the end of the main chain can better simulate these dynamic changes. This enhances the model's robustness against structural uncertainties in practical applications. When compared to the main-chain noise strategy, the main chain of a protein usually contains regions of high certainty and low certainty. The core regions that primarily determine protein function are often located in the non-terminal parts of the main chain. Therefore, to avoid excessive perturbation of the core structural areas of the protein, we opted to add noise solely to the terminal regions. The standard deviation of $0.02\,\text{Å}$ was selected as it effectively enhances the model's robustness, a strategy referenced from the work of Dauparas et al.In this experiment, we added Gaussian noise with a length of 0.02 times the sequence length. Results indicate (Table 6) that the model trained without noise performs well on the training set but lags behind the noise-added model on the test set. Our noise addition method performs the best, improving the sequence recovery rate by 0.55% compared to Ingraham[17] and by 3.84% compared to Dauparas[3]. We speculate that introducing noise into the main-chain might disrupt the existing backbone structure, while adding all-zero noise does not capture the disordered motion of the protein structure.To further evaluate the robustness of the proposed terminal noise scheme, we conducted a sensitivity analysis on the noise scale and the noise coverage ratio across the CATH4.2, CATH4.3, and $\text{Model}_{\text{final}}$ datasets. As shown in (Figure 4) the best performance was consistently achieved when the noise scale and coverage ratio were both set to 0.02. When either parameter was varied while keeping the other fixed, the sequence recovery rate decreased and the perplexity increased to different extents. This trend indicates that the benefit of terminal Gaussian noise does not increase monotonically with stronger perturbation or broader coverage. Instead, the performance gain depends on a balanced perturbation regime, in which the noise is sufficient to model the flexibility of terminal regions while remaining limited enough to avoid disrupting the structural signal. These results further support the rationality of the adopted setting in our experiments.

3.5 The effectiveness of size feature

We utilized the CASP14 test set to validate the effectiveness of the size feature proposed. We calculated the size feature following the steps outlined in the method section (Sec.2.3.4), and computed the average value of the features corresponding to each amino acid (Figure 3). Comparing panels (a) and (b), we observe that, despite Glycine and Serine having similar Δ_{NCO}

Table 6 The results of applying different noise addition methods to the Model$_{final}$ training, validation, and test sets.

	Clarification	Recovery / % (↑)			Perplexity (↓)		
		Train	Valid	Test	Train	Valid	Test
1	Not adding any noise	66.43%	53.75%	54.03%	3.14	4.17	4.14
2	The method by Ingraham	65.96%	58.03%	58.29%	3.03	3.73	3.72
3	The method by Dauparas	62.67%	56.93%	57.34%	3.34	3.84	3.83
4	Combining Ingraham and Dauparas	65.21%	58.00%	58.41%	3.17	3.74	3.71
5	Our method	**66.51%**	**59.06%**	**59.42%**	**3.02**	**3.59**	**3.56**

Underlined values denote the best results among the baseline methods, excluding our method. Higher Recovery and lower Perplexity indicate better performance.

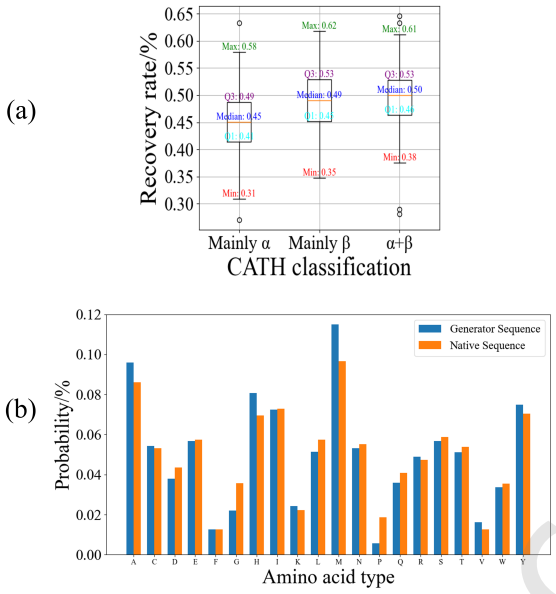

Fig. 3 The average value plots of residue size features. We conducted a statistical analysis of all residue size feature in the CASP14 test set and calculated the average value for each amino acid. (a): The average size of the Δ_{NCO} area formed by the main-chain atoms N, C, and O. (b): The average distance from the main-chain atom C_α to the Δ_{NCO} area.

areas, there remains a noticeable difference in the C_α to surface Δ_{NCO} distance. Similarly, Lysine and Proline exhibit similar C_α to surface Δ_{NCO} distances, yet their Δ_{NCO} areas show significant differences. Similarly, we can also clearly observe the differences in the area and distance features for other amino acids. Due to the differences in size feature among different amino acids, we introduced residue size feature into the model to enhance its ability to recognize, thereby improving model performance. The results of the feature ablation experiment show that after introducing this feature, the sequence recovery rate of the model improved by 1.2% (Experiment 4 in Table 7), validating the effectiveness of incorporating size features.

3.6 Comparison of model prediction speed

We further evaluated our model along with other methods (GCA[37], ProteinMPNN[3], PiFold[18], and GradeIF[42]) on the CATH 4.2,CATH 4.3 datasets and Model$_{final}$ datasets during the training process.We plotted the sequence recovery rates against the iteration process and training time (Figures 5 and 6). Overall,the results consistently indicate that our model exhibits a stable upward trend, demonstrating not only competitive accuracy but also robustness across different datasets.This suggests that the improvements are not confined to a single benchmark but generalize well across diverse evaluation settings.Figure 5 shows that our model outperforms other models in terms of sequence recovery rate on both the Model$_{final}$ and CATH4.2 datasets,achieving the best performance around the 45th epoch.On the CATH4.3 test set,although our model's performance was initially lower than that of PiFold[18] between the 34th and 41st epochs,it subsequently achieved a turnaround. Figure 6 shows that although our model's training speed is slower than some of the other models in the initial stages,it quickly surpasses the others and maintains a faster training rate.Both figures demonstrate that our model combines excellent performance with rapid convergence.

3.7 Comparison between generated sequences and natural sequences

We randomly selected 200 sequences from the Model$_{final}$ test set for comparison. These structures encompass three major CATH categories.The sequence recovery rates for different CATH categories are illustrated in Figure 7.The Pearson correlation coefficient between generated and natural sequences in terms of amino acid type composition is 0.972, indicating a high similarity in amino acid composition between the two. We also presented the analysis results of PiFold[18] and

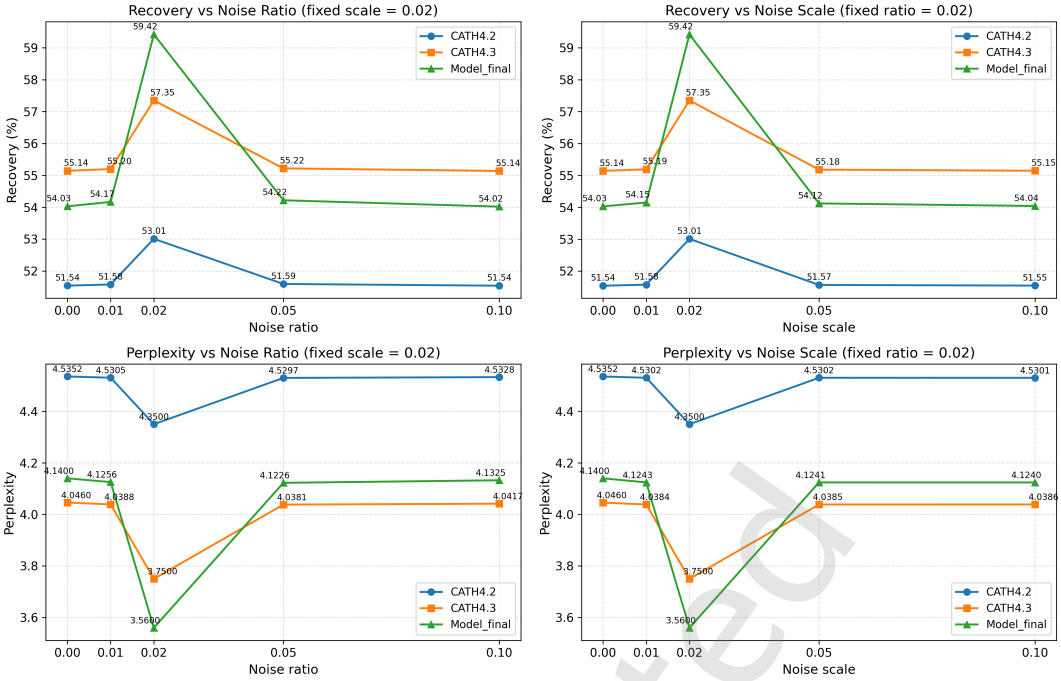

Fig. 4 The left column shows the effects of varying the noise coverage ratio with the noise scale fixed at 0.02, and the right column shows the effects of varying the noise scale with the noise coverage ratio fixed at 0.02. The top row reports sequence recovery, and the bottom row reports perplexity.

GradeIF[42] on the aforementioned 200 sequences (Figure 8, 9).The Pearson correlation coefficient for PiFold[18] is 0.957, and for GradeIF[42] it is 0.967. High similarity means that the generated sequence can fold into the given structure while also possessing the characteristics of a natural sequence.However,it is worth noting that,despite the overall high similarity, the frequency of certain amino acid types such as leucine and glutamic has increased in the generated sequences compared to natural sequences,while the usage of side-chain types such as glutamine and arginine has significantly decreased in the generated sequences,reflecting a preference for certain amino acids during the model generated process.

3.8 Model prediction confidence analysis

We present a scatter plot showing the relationship between the confidence and accuracy of the sequences generated by the model (Figure 10). The points in the scatter plot that are closer to the top right corner indicate that our model shows higher confidence and accuracy in generating the sequences.

3.9 Feature ablation

To investigate the specific contributions of different features to sequence recovery, we designed and conducted a series of ablation experiments (Table 7). Overall, edge features were found to be more important than node features, suggesting that the model may rely more on the relationships between residues when generating amino acid types. The size feature designed in our study contributed 1.20% to the recovery rate, demonstrating its effectiveness. The contribution of relative positional feature to the recovery rate was particularly significant, indicating a close correlation between the type of amino acid and the positions of surrounding amino acids. However, considering that the dimensionality of relative position features is proportional to the square of the number of atoms, introducing too many virtual atoms may lead to feature explosion and redundancy, which requires careful consideration during model design.

3.10 Comparison of prediction sequences from various methods

We compared the sequence identity between sequences predicted by different methods for the 200 sequences mentioned above (Figure 11). The figure shows that our model and the PiFold[19] model have a higher sequence identity, while the GradeIF[42] model has a lower sequence identity compared to the other methods.

Table 7 Results of model feature ablation experiments.All results were obtained by removing the corresponding features on the Model$_{final}$ dataset.

	Clarification	Recovery / % (↑)	Perplexity (↓)	Dedicate / %
Node feature	Angle feature	58.86%	3.65	-0.56%
	Spatial positional feature	59.08%	3.60	-0.34%
Edge feature	Spatial positional feature	58.37%	3.68	-1.05%
	Relative positional feature	51.49%	4.82	-7.93%
	Residue size feature	58.22%	3.70	-1.20%
	Intact model	**59.42%**	**3.56**	–

* The "–" symbol in the *Dedicate / %* column indicates the intact (complete) model, serving as the baseline reference. Other values represent the decrease in recovery rate caused by removing the corresponding feature from the intact model.

Table 8 The results of model ablation experiments. These results were obtained by using only different encoders and decoders under the optimal conditions of the Model$_{final}$ dataset.

	Clarification	**Recovery / % (↑)**	**Perplexity (↓)**	**Dedicate / %**
1	GAT	50.88%	4.65	-8.54%
2	GCN	45.92%	5.55	-13.50%
3	Single-Transformer	53.72%	4.24	-5.70%
4	MPNN	57.02%	3.88	-2.40%
5	Our model	**59.42%**	**3.56**	–

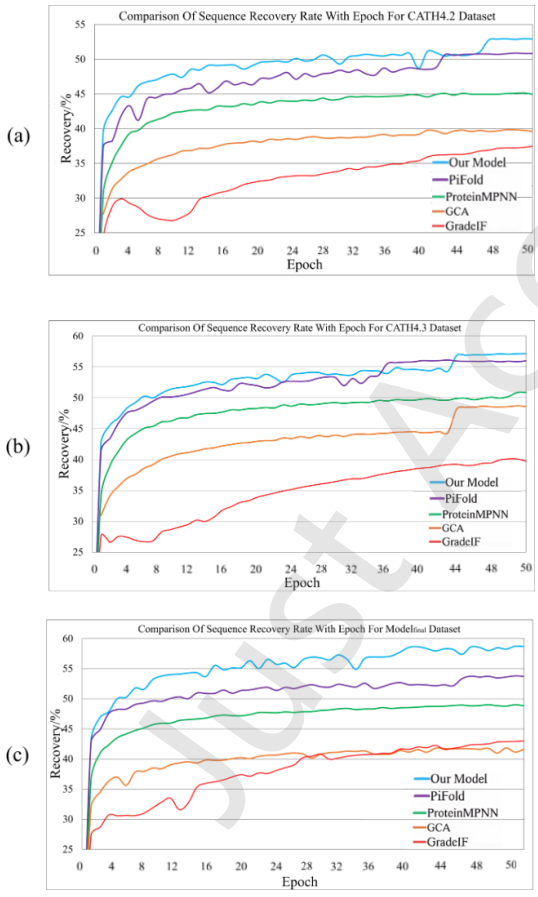

Fig. 5 Plot of sequence recovery with epoch for different models on dataset. Panels (a), (b), and (c) present the comparison results for the model on the CATH4.2, CATH4.3, and Model$_{final}$ datasets, respectively.

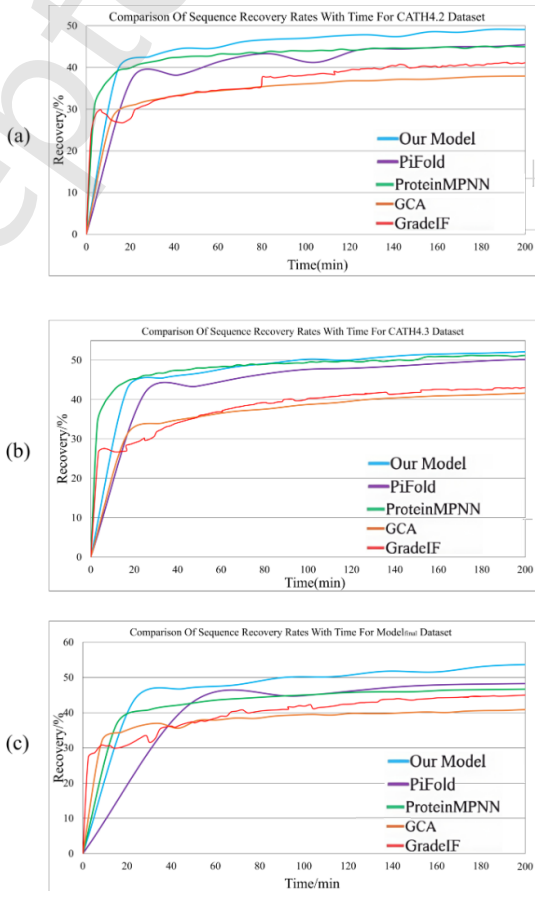

Fig. 6 Plot of sequence recovery over time for different models on the dataset. Panels (a), (b), and (c) present the comparison results for the model on the CATH4.2, CATH4.3, and Model$_{final}$ datasets, respectively.

3.11 Model ablation

We have meticulously designed the node features update structure and the edge features update structure, which together form the encoder-decoder of our model (MixInfo). Here, we utilize graph attention models (GAT[39]) , graph convolutional

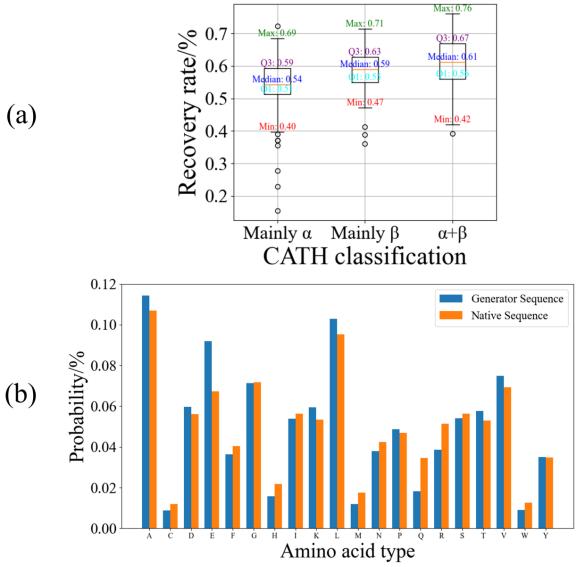

Fig. 7 Analysis of 200 sequences from the Modelfinal test set. (a) Boxplots of sequence recovery across three CATH structural classes. (b) Amino acid type distributions in generated and natural sequences.

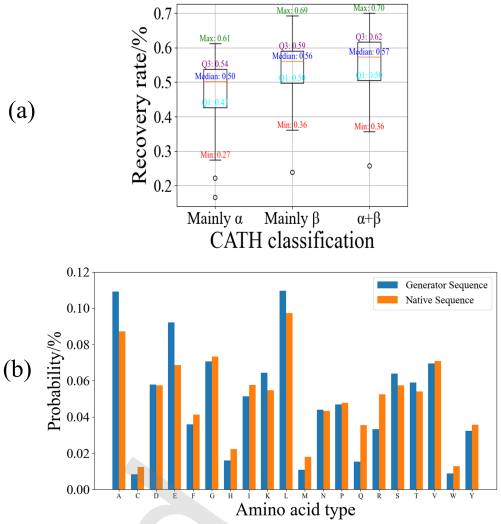

Fig. 8 Analysis of the 200 sampled sequences using PiFold. The legend is the same as in Figure 7.

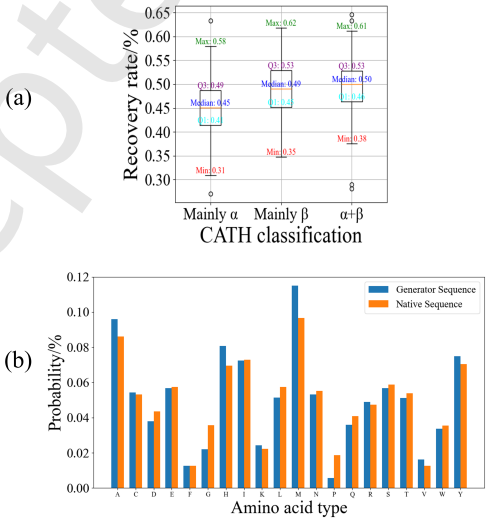

Fig. 9 Analysis of the 200 sampled sequences using GradeIF. The legend is the same as in Figure 7.

networks (GCN[43]), and single-layer Transformers based on QKV computation for attention from models like GraphTrans[17] to replace the MixInfo. In Experiment 4, we attempt to use a message passing network (MPNN) designed based on MLP[3] as the method for updating edge features and node features. In the MPNN, messages are designed to be transmitted multiple times within edge features and node features. The ablation results are shown in Table 8. The results of the graph convolution and graph attention models were relatively poor. We speculate that these models heavily rely on the graph's structure for aggregating neighboring nodes, which may hinder their ability to capture correlations between nodes that are close in the graph structure but distant in the sequence. MPNN performs well, indicating that simple iterative updates of edge features and node features can still achieve good results in sequence generation tasks.

3.12 Predict the generated sequences

We further analyzed the generated sequences by comparing different protein folding predictions. We selected three different types of prediction targets from the aforementioned 200 structures of natural proteins: 1iom (A chain), 4ofq (A chain), and 4i7w (A chain). We used AlphaFold2[40] to predict the structures of the sequences generated by our model. For each structure, we compared the predicted results (in blue) with the structures of the natural protein sequences (in red). The results indicate that our model achieves high recovery rates and low root mean square deviation (RMSD) across different CATH structure classifications, demonstrating a high degree of similarity to the actual structures.The one-dimensional sequence is the most crucial feature in protein-related research, as many higher-dimensional features are derived from it. For instance, the Position-Specific Scoring Matrix (PSSM) and the Hidden Markov Model (HHM) are both based on the one-dimensional sequence. Moreover, numerous pre-trained protein prediction models rely on one-dimensional sequence input. For example, the

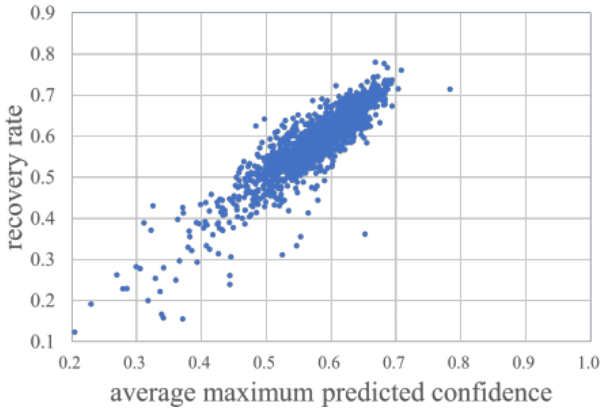

Fig. 10 Prediction confidence analysis. Recovery rate as a function of the average maximum predicted score (from 1433 structures in the Model$_{final}$ test dataset).

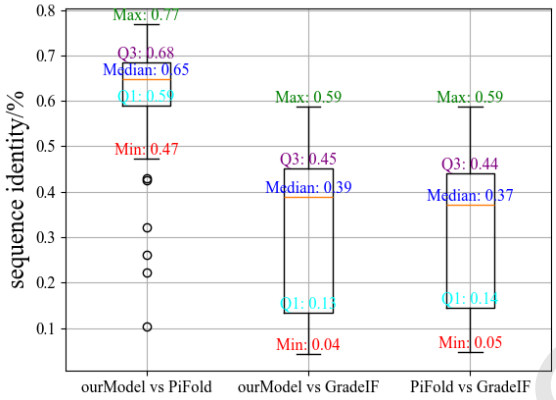

Fig. 11 Comparison of predicted sequences between methods. The average sequence consistency of our model with the PiFold model is 0.65, while the average with GradeIF is only 0.39.

ESM (Evolutionary Scale Model) can identify conserved regions of a protein chain based on its one-dimensional sequence. Therefore, the sequence predictions obtained by our model can be readily utilized for subsequent studies that require sequences as input. The high generation speed and low perplexity of our model constitute our strengths.

4 Discussion

Models with lower perplexity (higher confidence) offer distinct advantages across multiple stages of protein design. First, in de novo design scenarios without natural templates, sequence recovery provides limited guidance, whereas low perplexity serves as an effective indicator of internal sequence consistency and plausibility [48]. Second, during early exploration under constrained experimental resources, generating a compact yet reliable set of candidate sequences is critical, and low-perplexity predictions can substantially reduce downstream validation costs [49]. Third, in diversity-oriented sequence generation tasks, low-perplexity models facilitate the exploration of novel and credible sequence variants, whereas high-recovery models may bias toward natural sequences and thus limit diversity[50]. Fourth, in redesign exploration, where existing scaffolds are systematically modified to probe structural or functional variants, low-perplexity models such as MixInfoFold can prioritize modifications that maintain both sequence plausibility and structural integrity, thereby enhancing the efficiency of iterative design cycles. Collectively, these considerations indicate that low-perplexity models can complement high-recovery approaches by providing more reliable candidate sequences in contexts where confidence and adherence to constraints are prioritized.

5 Model algorithm

Algorithm 1 MixInfoFold Structure

Input: k-nearest neighbor residue sets: $E_{idx} \in \mathbb{R}^{B\times 1.02S\times n}$
angle feature: $V_{angle} \in \mathbb{R}^{B\times 1.02S\times 12}$
spatial position feature: $V_{emb} \in \mathbb{R}^{B\times 1.02S\times 9}$, $E_{emb} \in \mathbb{R}^{B\times 1.02S\times n\times 7}$
relative position feature: $P_{emb} \in \mathbb{R}^{B\times 1.02S\times 736}$
size feature: $R_{size} \in \mathbb{R}^{B\times 1.02S\times n\times 32}$
real sequence: $S_{true} \in \mathbb{R}^{B\times 1.02S}$
mask matrix: $M \in \mathbb{R}^{B\times 1.02S}$

Output: generate sequence: $S \in \mathbb{R}^{B\times 1.02S}$

// packaging edge features and node features
$V_n \Leftarrow W_v concat(V_{angle}, V_{emb}) \in \mathbb{R}^{B\times 1.02S\times hiddim}$ ▷ node features
$E_n \Leftarrow W_e concat(E_{emb}, P_{emb}, R_{size}) \in \mathbb{R}^{B\times 1.02S\times n\times hiddim}$ ▷ edge features
// encoder
for $i = 0$ to 3 **do**
  $VE1 \Leftarrow concat(E_n, E_{idx}(V_n)) \in \mathbb{R}^{B\times 1.02S\times n\times 2hiddim}$
  // the first attention update
  $V_n attention1 \Leftarrow Attention(V_n, VE1, E_{idx}, M) \in \mathbb{R}^{B\times 1.02S\times hiddim}$
  // residual and normalization
  $V_n 1 \Leftarrow Norm(Dropout(V_n attention1) + V_n) \in \mathbb{R}^{B\times 1.02S\times hiddim}$
  $VE2 \Leftarrow concat(E_n, E_{idx}(V_n 1)) \in$

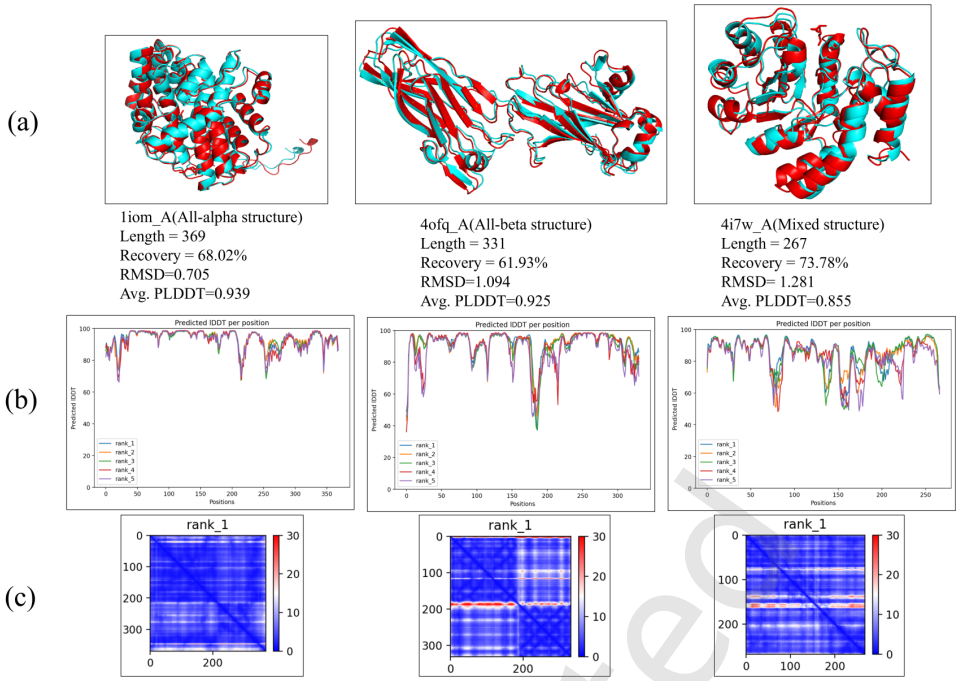

Fig. 12 Performing structural predictions on designed sequences using AlphaFold2. (a). Sequences predicted by AlphaFold2: where red represents natural protein sequences and blue represents sequences predicted by the model. (b). PLDDT scores for each residue position in the sequence, where residues with pLDDT $\geq$ 90 exhibit very high model confidence, residues with 90 $>$ pLDDT $\geq$ 70 are categorized as confident, residues with 70 $>$ pLDDT $\geq$ 50 have low confidence, and residues with pLDDT $<$ 50 correspond to very low confidence. (c). Predicted alignment error (PAE) plot of the sequence. In the PAE plot, each point represents a residue, and the color of the point indicates the prediction accuracy of that residue (darker blue indicates higher accuracy, while darker red indicates lower accuracy).

$\mathbb{R}^{B\times1.02S\times n\times hiddim}$

// the second attention update

$E_n attention \Leftarrow Attention(V_n1, VE2, E_{idx}, M) \in \mathbb{R}^{B\times1.02S\times n\times hiddim}$

$E_n1 \Leftarrow Norm(Dropout(E_n attention) + E_n) \in \mathbb{R}^{B\times1.02S\times n\times hiddim}$

$E_n2 \Leftarrow PositionWiseFeedForward(E_n1) \in \mathbb{R}^{B\times1.02S\times n\times hiddim}$

$E_n^{'} \Leftarrow Norm(Dropout(E_n2) + E_n1) \in \mathbb{R}^{B\times1.02S\times n\times hiddim}$

$VE3 \Leftarrow concat(E_n^{'}, E_{idx}(V_n1)) \in \mathbb{R}^{B\times1.02S\times n\times 2hiddim}$

// the third attention update

$V_n attention2 \Leftarrow Attention(V_n1, VE3, E_{idx}, M) \in \mathbb{R}^{B\times1.02S\times hiddim}$

$V_n2 \Leftarrow Norm(Dropout(V_n attention2) + V_n1) \in \mathbb{R}^{B\times1.02S\times hiddim}$

$V_n3 \Leftarrow PositionWiseFeedForward(V_n2) \in \mathbb{R}^{B\times1.02S\times hiddim}$

$V_n^{'} \Leftarrow Norm(Dropout(V_n3) + V_n2) \in \mathbb{R}^{B\times1.02S\times hiddim}$

$V_n = V_n^{'} + V_n \in \mathbb{R}^{B\times1.02S\times hiddim}, E_n = E_n^{'} + E_n \in \mathbb{R}^{B\times1.02S\times n\times hiddim}$

end for

// sequential decoding method

$ES_{true} \Leftarrow concat(E_n, E_{idx}(S_{true})) \in \mathbb{R}^{B\times1.02S\times n\times 2hiddim}$

$E_{zero} \Leftarrow concat(E_n, E_{idx}(zero\_like(S_{true}))) \in \mathbb{R}^{B\times1.02S\times n\times 2hiddim}$

$VE_{zero} \Leftarrow concat(E_{zero}, E_{idx}(V_n))) \in \mathbb{R}^{B\times1.02S\times n\times 2hiddim}$

$M_{attend} \Leftarrow E_{idx}(M) \in \mathbb{R}^{B\times1.02S\times n\times 1}$

// decoder

for $i = 0$ to 3 **do**

$VES_{true} \Leftarrow concat(ES_{true}, E_{idx}(V_n)) \in \mathbb{R}^{B\times1.02S\times n\times 3hiddim}$

$VES \Leftarrow M_{attend} * VES_{true} + (1 - M_{attend}) * VE_{zero} \in \mathbb{R}^{B\times1.02S\times n\times 3hiddim}$

// MLP update

$E_n1 \Leftarrow MLP(VES) \in \mathbb{R}^{B\times1.02S\times n\times 3hiddim}$

$E_n^{'} \Leftarrow Norm(Dropout(E_n1) + VES) \in \mathbb{R}^{B\times1.02S\times n\times 3hiddim}$

// the fourth attention update

$V_n attention \Leftarrow$

$Attention(V_n, E_n^{'}, E_{idx}, M) \in \mathbb{R}^{B\times 1.02S\times hiddim}$

$V_n1 \Leftarrow Norm(Dropout(V_nattention)+V_n) \in \mathbb{R}^{B\times 1.02S\times hiddim}$

$V_n2 \Leftarrow PositionWiseFeedForward(V_n1) \in \mathbb{R}^{B\times 1.02S\times hiddim}$

$V_n^{'} \Leftarrow Norm(Dropout(V_n2) + V_n1) \in \mathbb{R}^{B\times 1.02S\times hiddim}$

$V_n = V_n^{'} + V_n \in \mathbb{R}^{B\times 1.02S\times hiddim}$

end for

$S \Leftarrow log\_softmax(W_{out} * V_n) \in \mathbb{R}^{B\times 1.02S}$

Supplementary Algorithm 1 — MixInfoFold Structure. MixInfoFold consists of three layers of encoders and decoders (MixInfo). The feature dimensions include batch size (B), sequence length (S), number of nearest neighbors (n), and hidden layer dimension (hiddim). The input vectors are all injected with Gaussian noise equal to 0.02 times the sequence length, and a mask vector is used for identification. We combine the feature vectors into edge features and node features, iteratively calculating the attention of edge and node features in MixInfo to complete the feature updates. The sequential decoding method incorporates true sequence information into the edge features of the decoder. The output vector is the sequence generated by the model.

Algorithm 2 Attention Structure

Input: node features: $V_n \in \mathbb{R}^{B\times 1.02S\times hiddim}$
edge features: $E_n \in \mathbb{R}^{B\times 1.02S\times n\times 2hiddim}$
k-nearest neighbor residue sets: $E_{idx} \in \mathbb{R}^{B\times 1.02S\times n}$
mask matrix: $M \in \mathbb{R}^{B\times 1.02S}$

Output: node features: $V_n \in \mathbb{R}^{B\times 1.02S\times hiddim}$ **or** edge feature: $E_n \in \mathbb{R}^{B\times 1.02S\times hiddim}$

if *encoder* **then**
$VE \Leftarrow concat(E_n, E_{idx}(V_n)) \in \mathbb{R}^{B\times 1.02S\times n\times 3hiddim}$
$weight \Leftarrow MLP(VE) \in \mathbb{R}^{B\times 1.02S\times n\times n\_heads\times 1}$
$values \Leftarrow MLP(E) \in \mathbb{R}^{B\times 1.02S\times n\times n\_heads\times d}$
$attend_{logits} \Leftarrow \frac{\text{weight}}{\sqrt{\text{d}}} \in \mathbb{R}^{B\times 1.02S\times n\times n\_heads\times 1}$
else if *decoder* **then**
$queries \Leftarrow W_q(V_n) \in \mathbb{R}^{B\times 1.02S\times 1\times n\_heads\times 1\times d}$
$keys \Leftarrow W_k(E_n) \in \mathbb{R}^{B\times 1.02S\times n\times n\_heads\times d\times 1}$
$values \Leftarrow W_v(E_n) \in \mathbb{R}^{B\times 1.02S\times n\times n\_heads\times d}$
$attend_{logits} \Leftarrow \frac{\text{queries}*\text{keys}}{\sqrt{\text{d}}} \in \mathbb{R}^{B\times 1.02S\times n\_heads\times n}$
end if
$attend \Leftarrow M * softmax(attend_{logits}) \in \mathbb{R}^{B\times 1.02S\times n\times n\_heads\times 1}$
if Output $== E_n$ **then**
$E_n \Leftarrow attend * values \in \mathbb{R}^{B\times 1.02S\times n\times hiddim}$
else if Output $== V_n$ **then**
$V_n \Leftarrow sum(attend*values) \in \mathbb{R}^{B\times 1.02S\times hiddim}$
end if

Supplementary Algorithm 2 — Attention Structure. The feature dimensions of the Attention structure include batch size (B), sequence length (S), number of nearest neighbors (n), hidden layer dimension (hiddim), number of attention heads (n_heads), and dimension of each attention head (d). The input vectors include: node features, edge features, nearest neighbor matrix, and mask matrix. Depending on the updated features, the output vector can be either node features or edge features. It is important to note that the calculation of attention in the encoder and decoder differs because the edge features in the decoder contain information from the true sequence (sequential decoding method). If we combine the edge features and node features into a single vector to compute the attention weights, it would cause the updated node features to directly output the true sequence, leading to the failure of the generation task.

6 Conclusion

In this paper, we propose a novel generative model called MixInfoFold for sequence prediction based on protein three-dimensional structure. The model combines four features, including size feature we designed, and leverages three-dimensional structural information to generate corresponding sequence. Meantime, we introduce a unique noise handling method to enhance the model's performance. Compared to Rosetta, our model not only achieves higher accuracy but also requires much less computational time for solving sequence generation problems. Compared to other latest models, our model achieves better performance in sequence generation. We used multiple methods to validate the rationality and feasibility of the sequences generated by our model.

7 Code and Data availability

Our code is open to all people and you can find it at: `https://github.com/IMCTGD/MixInfoFold`.

The following is the source of the dataset used in our article: CATH4.2 Dataset: `https://dl.acm.org/doi/10.5555/3454287.3455704`. CATH4.3 Dataset:

https://www.biorxiv.org/content/10.1101/2022.04.10.487779v1. Model$_{final}$ Dataset: https://www.nature.com/articles/s43588-022-00273-6. TS50 test set: https://pubmed.ncbi.nlm.nih.gov/24898915/. CASP14 test set: https://www.predictioncenter.org/casp14/index.cgi.

We have cited their article in the paper as requested by the original authors, and once again, we express our heartfelt gratitude to them.

References

[1] Pearson, W. R. and M. L. Sierk. The limits of protein sequence comparison? *15(3): 254-260,* 2005 **Current opinion in structural biology**.

[2] Dahiyat, B. I. and S. L. Mayo. De novo protein design: fully automated sequence selection. *278(5335): 82-87,* 1997. **Science**.

[3] Dauparas, Justas Anishchenko, Ivan Bennett, Nathaniel Bai, Hua Ragotte, Robert J Milles, et al. Robust deep learning–based protein sequence design using ProteinMPNN. *378(6615): 49-56*, 2022. **Science**.

[4] Leaver-Fay, A., M. Tyka, S. M. Lewis. (2011). ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. *487:545-574.* **Methods in enzymology, Elsevier**.

[5] Simonson, T., T. Gaillard, D. Mignon. Computational protein design: the Proteus software and selected applications. *34(28):2472-2484*, 2013. **Journal of computational chemistry**.

[6] Huang, X., R. Pearce and Y. Zhang. EvoEF2: accurate and fast energy function for computational protein design. *36(4):1135-1142*, 2020. **Bioinformatics**.

[7] Xiong, P., M. Wang, X. Zhou. Protein design with a comprehensive statistical energy function and boosted by experimental selection for foldability. *5(1):5330*, 2014. **Nature communications**.

[8] Xiong, P., X. Hu, B. Huang. Increasing the efficiency and accuracy of the ABACUS protein sequence design method. *36(1):136-144*, 2020.**Bioinformatics**.

[9] Zhou, J., A. E. Panaitiu and G. Grigoryan. A general-purpose protein design framework based on mining sequence–structure relationships in known protein structures. *117(2):1059-1068*, 2020. **Proceedings of the National Academy of Sciences**.

[10] Huang, Po-Ssu, Baker, David, Boyken, Scott, et al. The coming of age of de novo protein design. *537(7620): 320*, 2016. **Nature**.

[11] Siegel, J. B., A. Zanghellini, H. M. Lovick. Computational Design of an Enzyme Catalyst for a Stereoselective Bimolecular Diels-Alder Reaction. *329(5989):309-313*, 2010. **Science**.

[12] Bale, J. B., S. Gonen, Y. Liu. Accurate design of megadalton-scale two-component icosahedral protein complexes. *(6297), 2016*. **Science**

[13] Balakrishnan, S., H. Kamisetty, J. G. Carbonell. Learning generative models for protein fold families. *79(4): 1061-1078*, **Proteins: Structure, Function, and Bioinformatics** 2011.

[14] Marks, D. S., L. J. Colwell, R. Sheridan. Protein 3D structure computed from evolutionary sequence variation. *6(12): e28766*, 2011. **PloS one**.

[15] Morcos, F., A. Pagnani, B. Lunt. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *108(49): E1293-E1301*, 2011 **Proceedings of the National Academy of Sciences**.

[16] Vaswani, Ashish Shazeer, Noam Parmar, Niki Uszkoreit, Jakob Jones, Llion Gomez. Attention is all you need. *30*, 2017 **Advances in neural information processing systems**.

[17] John Ingraham, Vikas K Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. *2019* **Neural Information Processing Systems**.

[18] Gao, Z., C. Tan, P. Chacón. PiFold: Toward effective and efficient protein inverse folding. *2023* **International Conference on Learning Representations**.

[19] Li, Z., Y. Yang, E. Faraggi. Direct prediction of profiles of sequences compatible with a protein structure by neural networks with fragment-based local and energy-based nonlocal profiles. *82(10): 2565-2573*, 2014 **Proteins: Structure, Function, and Bioinformatics**.

[20] Yang, Y. and Y. Zhou. Ab initio folding of terminal segments with secondary structures reveals the fine difference between two closely related all-atom statistical energy functions. *17(7): 1212-1219*, 2008. **Protein science**.

[21] Wang, J., H. Cao, J. Z. Zhang. Computational protein design with deep learning neural networks. *8(1): 1-9*, 2018. **Scientific reports**.

[22] O'Connell, J., Z. Li, J. Hanson. SPIN2: Predicting sequence profiles from protein structures using deep neural networks. *86(6):629-633*, 2018. **Proteins: Structure, Function, and Bioinformatics**.

[23] Torng, W. and R. B. Altman. 3D deep convolutional neural networks for amino acid environment similarity analysis. *18: 1-23, 2017*. **BMC bioinformatics**.

[24] Wang, Y., F. Tian, D. He. on-autoregressive machine translation with auxiliary regularization. **Proceedings of the AAAI conference on artificial intelligence**. 2019.

[25] Zhang, Y., Y. Chen, C. Wang. Prodconn-protein design using a convolutional neural network. *118(3):43a-44a, 2020.***Biophysical Journal.**

[26] Qi, Y. and J. Z. Zhang. DenseCPD: improving the accuracy of neural-network-based computational protein sequence design with DenseNet. *60(3): 1245-1252*, 2020. **Journal of chemical information and modeling**.

[27] Norn, C., B. I. Wicky, D. Juergens. Protein sequence design by conformational landscape optimization. *118(11): e2017228118*, 2021. **Proceedings of the National Academy of Sciences**.

[28] Yang, J., I. Anishchenko, H. Park. Improved protein structure prediction using predicted interresidue orientations. *117(3): 1496-1503, 2020.* **Proceedings of the National Academy of Sciences**.

[29] Kipf, T. N. and M. Welling. Semi-supervised classification with graph convolutional networks. *:1609.02907, 2016.* **arXiv preprint**.

[30] Wang, X., S. T. Flannery and D. Kihara. Protein docking model evaluation by graph neural networks. *8: 647915*, 2021. **Frontiers in Molecular Biosciences,**

[31] Orellana, G. A., J. Caceres-Delpiano, R. Ibañez. Protein sequence sampling and prediction from structural data. *2021.2009. 2006.459171*, 2021. **bioRxiv**.

[32] Huynh, D. Q. Metrics for 3D rotations: Comparison and analysis. *35: 155-164*, 2009. **Journal of Mathematical Imaging and Vision**.

[33] Adams, P. D., R. W. Grosse-Kunstleve, L.-W. Hung. PHENIX: building new software for automated crystallographic structure determination. *58(11): 1948-1954, 2002.* **Acta Crystallographica Section D: Biological Crystallography**.

[34] Hsu, C., R. Verkuil, J. Liu. Learning inverse folding from millions of predicted structures. *2022*. **International conference on machine learning PMLR**.

[35] Liu, Yufeng Zhang, Lu Wang, Weilun Zhu, Min Wang, Chenchen Li, et al. Rotamer-free protein sequence design based on deep learning and self-consistency. *2(7): 451-462, 2022.* Nature Computational Science,

[36] Das, R. and D. Baker. Macromolecular modeling with rosetta. *77: 363-382, 2008.* **Annu. Rev. Biochem**.

[37] Tan, Cheng Gao, Zhangyang Xia, Jun Hu, Bozhen Li, Stan Z. Generative de novo protein design with global context. *2204.10673, 2022*. **arXiv preprint arXiv**.

[38] Yi, K., B. Zhou, Y. Shen, et al. Graph denoising diffusion for inverse protein folding. *36 2024*. **Advances in Neural Information Processing Systems**.

[39] Veličković, Petar Cucurull, Guillem Casanova, Arantxa Romero, Adriana Lio, Pietro Bengio, Yoshua. Graph attention networks. *1710.10903, 2017*. **arXiv preprint arXiv**.

[40] Jumper, John Evans, Richard Pritzel, Alexander Green, Tim Figurnov, Michael Ronneberger, et al. Highly accurate protein structure prediction with AlphaFold. *596(7873):583-589, 2021*. **Nature**

[41] Kipf, Thomas N. , and M. Welling. Semi-Supervised Classification with Graph Convolutional Networks. *2017.* **International Conference on Learning Representations**.

[42] Jingxue Wang, Huali Cao, John ZH Zhang, and Yifei Qi. Computational protein design with deep learning neural networks. *8(1):1–9, 2018.* **Scientific reports**.

[43] Sheng Chen, Zhe Sun, Lihua Lin, Zifeng Liu, Xun Liu, Yutian Chong, Yutong Lu, Huiying Zhao, and Yuedong Yang. To improve protein sequence profile prediction through image captioning on pairwise residue distance map. **Journal of chemical information and modeling**.

[44] Lu, A. X., Yan, W., Robinson, S. A., Yang, K. K., Gligorijevic, V., Cho, K., Bonneau, R., Abbeel, P., & Frey, N Generating All-Atom Protein Structure from Sequence-Only Training Data. **bioRxiv, 2024–12. 2024. Cold Spring Harbor Laboratory.**.

[45] Zhang, C., Liu, X., Luo, Y., Song, S., & Peng, J An interpretable deep geometric learning model to predict the effect of mutation on protein–protein interaction binding affinity.**Journal of Cheminformatics, 2025-03-21.**.

[46] Atkinson, T., Barrett, T. D., Cameron, S., Guloglu, B., Greenig, M., Tan, C. B., Robinson, L., Graves, A., Copoiu, L., & Laterre, A Protein sequence modelling with Bayesian flow networks. **Nature Communications, 16(1), 3197.**.

[47] Rose, T., Monti, N., Anand, N., & Shen, T Harnessing pre-trained models for accurate prediction of protein–ligand binding affinity. **BMC Bioinformatics, 26(1), 6064.**.

[48] I. Anishchenko et al. "De novo protein design by deep network hallucination." **Nature, 600(7889), 547–552, Dec. 2021, doi: 10.1038/s41586-021-04184-w.**

[49] J. Yu, J. Mu, T. Wei, and H.-F. Chen "Multi-indicator comparative evaluation for deep learning-based protein sequence design methods." **Bioinformatics, 40(2), Jan. 2024, doi: 10.1093/bioinformatics/btae037.**

[50] F. Ye et al. "ProteinBench: A Holistic Evaluation of Protein Foundation Models." **arXiv, 2024, doi: 10.48550/ARXIV.2409.06744.**

An Zeng Received a Doctorate in Computer Application Technology from South China University of Technology in July 2005. Secretary-General of the Guangdong Provincial Industrial Software Society, Member of the Chinese Society of Biomedical Engineering, Director of the Guangdong Provincial Biomedical Engineering Society, Member of the CCF Collaborative Computing Committee, Corresponding Member of the CCF Artificial Intelligence Committee, Member of the Bioinformatics and Artificial Life Professional Committee of the Chinese Association for Artificial Intelligence, and Vice President of the Guangzhou Computer Society.

CanHui Chen After receiving the Bachelor's degree in Engineering in 2024, began graduate studies at the School of Computer Science, Guangdong University of Technology. Current research interests include graph neural networks for protein analysis.

Jinrong Li Graduated with a Bachelor's degree in Engineering in 2018. Since 2022, has been studying at the School of Computer Science, Guangdong University of Technology, with research supported by the university. Current research interests include protein sequence design.

Baoyao Yang Since January 2021, Associate Professor in the Department of Computer Science and Technology, School of Computer Science, Guangdong University of Technology. Has published papers in several SCI Tier 1 journals. Research interests include machine learning and smart healthcare.

Dan Pan Received a Ph.D. in Computer Science from the Department of Computer Science, School of Electronics and Information, South China University of Technology in 2001. Member of the China Computer Federation, Member of the China Society of Image and Graphics, and Member of the Guangdong Provincial Hospital Association. Has served as a committee member of the Medical Robotics and Artificial Intelligence Branch of the Guangdong Provincial Biomedical Engineering Society, and a committee member of the Alzheimer's Disease Branch of the Guangdong Provincial Precision Medicine Application Society. Research interests include artificial intelligence technology and its applications across various fields, including biomedical engineering.