Fold2Seq: A Joint Sequence(1D)-Fold(3D) Embedding-based Generative Model for Protein Design

Yue Cao [1 2 *] Payel Das [1] Vijil Chenthamarakshan [1] Pin-Yu Chen [1] Igor Melnyk [1] Yang Shen [2]

. .
.

Abstract

Designing novel protein sequences for a desired 3D topological fold is a fundamental yet non-trivial task in protein engineering. Challenges exist due to the complex sequence–fold relationship, as well as the difficulties to capture the diversity of the sequences (therefore structures and functions) within a fold. To overcome these challenges, we propose **Fold2Seq**, a novel transformer-based generative framework for designing protein sequences conditioned on a specific target fold. To model the complex sequence–structure relationship, Fold2Seq jointly learns a sequence embedding using a transformer and a fold embedding from the density of secondary structural elements in 3D voxels. On test sets with single, high-resolution and complete structure inputs for individual folds, our experiments demonstrate improved or comparable performance of Fold2Seq in terms of speed, coverage, and reliability for sequence design, when compared to existing state-of-the-art methods that include data-driven deep generative models and physics-based RosettaDesign. The unique advantages of fold-based Fold2Seq, in comparison to a structure-based deep model and RosettaDesign, become more evident on three additional real-world challenges originating from low-quality, incomplete, or ambiguous input structures. Source code and data are available at https://github.com/IBM/fold2seq.

1. Introduction

Computationally designing protein sequences that fold into desired 3D structures has a broad range of applications ranging from therapeutics to materials (Kraemer-Pecore et al., 2001). Despite significant advancements in methodologies and computing power, this task known as inverse protein design still remains challenging, primarily due to the vast size of the sequence space as well as the difficulty of learning a function that maps from the 3D structure space to the 1D sequence space.

While majority of data-driven approaches (Chen et al., 2019; O'Connell et al., 2018; Wang et al., 2018b; Ingraham et al., 2019; Strokach et al., 2020) are focusing on designing sequences for a desired backbone structure, only a few works (Greener et al., 2018; Karimi et al., 2020) have studied protein design for a desired fold. A protein fold is defined by the spatial arrangement (or topology) of its 3D form of local segments called secondary structure elements or SSEs (Hou et al., 2003). As protein structure is inherently hierarchical, a complete native structure can have fold combinations and a fold can be present in many protein structures. A simple fold or topological architecture can be highly adaptable, as shown by the low-sequence homology among its members, and the different functions they carry out (Basanta et al., 2020; Chandra et al., 2001; Boutemy et al., 2011). Therefore, a primary goal of *de novo* protein design is to generate a larger and more diverse set of protein structures than currently available yet still consistent with a specific fold, which has proven to be a means for achieving new functions through design (Basanta et al., 2020; Woolfson et al., 2015). In contrast, targeting a backbone structure *per se* is known to restrict the diversity and novelty of the designs, as "high-resolution protein backbone coordinates contain some memory of the original native sequence" (Kuhlman et al., 2003). Accordingly, an ensemble of structures is a better representative of a fold than a single structure, as it additionally captures the structural and therefore functional diversity within the fold.

Compared to structure-based protein design, fold-based protein design carries additional challenges: the difficul-

ties of learning a good fold representation for accurately capturing the diversity of the fold space and the complex fold-sequence relationship. Despite the impressive progress made by recent data-driven methods, aforementioned challenges are not fully solved. **First**, the current fold representation methods are either hand-designed, or constrained and do not capture the complete original fold space (Greener et al., 2018; Karimi et al., 2020; Koga et al., 2012), resulting in poor generalization or efficiency. **Second**, the (1D) sequence encoding and the (3D) fold encoding are learned separately in previous methods, which makes two latent domains heterogeneous. Such heterogeneity across two domains actually increases the difficulty of learning the complex sequence–fold relationship.

To fill the aforementioned gaps, the **main contributions** of this work are as follows:

1. We propose a novel fold representation, through first representing the 3D structure by the voxels of the SSE density, and then learning the fold representation through a transformer-based fold encoder. Compared to previous fold representations, this one has several advantages: it preserves the entire spatial information of SSEs in a scale-free manner, does not need any pre-defined rules, and can be automatically extracted from a given protein structure. The density model also loosens the structure rigidity so that the structural variation and missing information is better handled.

2. We employ a novel joint sequence–fold embedding learning framework into the transformer-based auto-encoder model. By learning a joint latent space between sequences and folds, **Fold2Seq**, mitigates the heterogeneity between two different domains and is able to better capture the sequence-fold relationship, as reflected in the results.

3. We develop several novel fold-level assessment metrics. Using those, we demonstrate that Fold2Seq has superior or comparable performance in perplexity, sequence recovery rate, and structure recovery rate, when compared to competing methods including the state-of-the-art RosettaDesign and other neural-net models on the benchmark test set. More importantly, Fold2Seq-generated sequences provide better coverage (diversity) within a specified fold. Ablation study shows that this improved performance can be directly attributed to our algorithmic innovations.

4. Experiments on real-world challenges comprised of low-resolution structures, structures with missing residues, and Nuclear Magnetic Resonance (NMR) ensembles further demonstrate the unique practical utility and versatility of Fold2Seq compared to the structure-based baselines.

2. Related Work

Data-driven Protein Design. A significant surge of protein design studies that deeply exploit the data through modern artificial intelligence algorithms has been witnessed in the last three years. There appear a gallery of methods that focus on design protein sequences conditioned on the *backbone structure* (Chen et al., 2019; O'Connell et al., 2018; Wang et al., 2018b). Recently, Strokach et al. (2020) formulated the inverse protein design as a constraint satisfaction problem (CSP) and applied the graph neural networks for generating protein sequences conditioned on the residue-residue distance map. Ingraham et al. (2019) developed a graph-based transformer for generating protein sequences conditioned on the either rigid or flexible protein backbone information. Nevertheless, there are only a few studies that investigated protein design conditioned directly on the *protein fold*. Greener et al. (2018) used the conditional variational autoencoder for generating protein sequences conditioned on a given fold. Karimi et al. (2020) developed a guided conditional Wasserstein Generative Adversarial Networks (gcWGAN) also for the fold-based design.

Protein Fold Representation. For an extensive overview of molecular representations, including those of proteins, please see David et al. (2020). Murzin et al. (1995) and Orengo et al. (1997) manually classified protein structures in a hierarchical manner based on their structural similarity, resulting into one-hot encoding of the fold representations. Taylor (2002) represents a protein fold using a "periodic table" that was later used for inverse fold design (Greener et al., 2018). However, it considers three pre-defined folds ($\alpha\beta\alpha$ layer, $\alpha\beta\beta\alpha$ layer and $\alpha\beta$ barrel) for defining a fold space, limiting the spatial information content of the fold significantly. Hou et al. (2003) chose hundreds of representative proteins and calculated the similarity scores among them. This similarity matrix was then converted into a distance matrix for kernel Principal Component Analysis (kPCA). A similar idea was used in Karimi et al. (2020) for inverse protein design. This representation needs a set (all-α, all-β, α/β and $\alpha+\beta$) of structures along with a similarity metric for defining a fold space, which may lead to biased or restricted representation of the fold space. Further, use of a similarity (or distance) matrix between fold pairs to learn fold representation, in principle, may not preserve the detailed spatial information of the fold. Finally, Koga et al. (2012) summarized three rules that describe the junctions between adjacent secondary structure elements for a specific fold. These rules are hand designed for a subset of structures, which makes the representation restricted to a small part of the fold space and offers limited generalizability during conditional sequence generation.

Joint Embedding Learning. Joint embedding learning across different data modalities was first proposed by Ngiam et al. (2011) on audio and video signals. Since then, such approaches have been widely used in cross modal retrieval or captioning (Arandjelovic & Zisserman, 2018; Gu et al., 2018; Peng & Qi, 2019; Chen et al., 2018; Wang et al.,
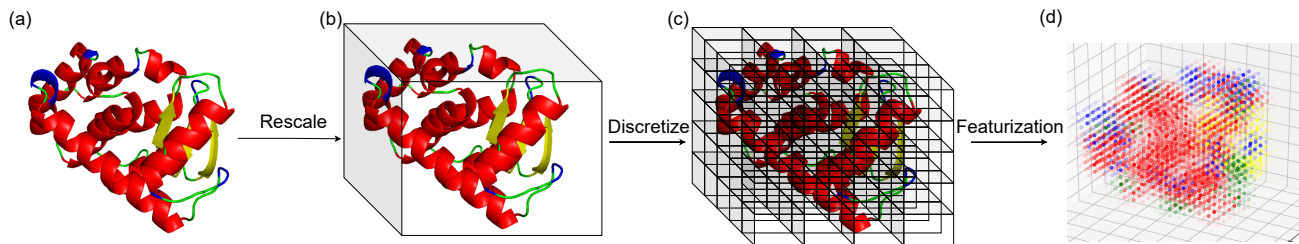
Figure 1: (a) The structure of T4 lysozyme (PDB ID 107L). The secondary structures are colored as: helices in red, beta sheets in yellow, loops in green and bend/turn in blue. (b) The structure is rescaled to fit the $40\text{Å} \times 40\text{Å} \times 40\text{Å}$ cubic box. (c) The box is discretized into voxels. (d) Features of each voxel are obtained from the structure content of the voxel.

2013; Dognin et al., 2019). In few/zero-shot learning, joint feature-label embedding was used (Zhang & Saligrama, 2016; Socher et al., 2013). Several studies have shown usefulness of learning joint embedding for single modal classification (Ngiam et al., 2011; Wang et al., 2018a; Toutanova et al., 2015). Moreover, Chen et al. (2018) used joint embedding learning for text to shape generation. Joint sequence–label embedding is also explored for or applied to molecular prediction/generation (Cao & Shen, 2021; Das et al., 2018).

3. Methods

3.1. Background

A protein consists of a linear chain of amino acids (residues) that defines its 1D sequence. Chemical composition and interactions with neighboring residues drive the folding of a sequence into different secondary structure elements or SSEs (helix, beta-sheet, loop, etc., see Fig. 1(a)), that eventually forms a complete native 3D structure. A **protein fold** captures the structural consensus of the 3D topology and the composition of those secondary structure elements.

3.2. Fold Representation through 3D voxels of the SSE density

In *de novo* protein design that we target, no backbone structure is assumed. Instead, a topological "blueprint" (consistent with the desired fold) is given. And initial backbone structures can be generated accordingly using fragment assemblies (Huang et al., 2016). In this study we focus on generating fold representations once the structures are available and additionally explore the challenges from such "blueprint" input structures through three real-world challenges.

We hereby describe how we represent the 3D structure to explicitly capture the fold information, as illustrated in Fig. 1. The position (3D coordinates) of each residue is represented by its α-carbon. For a given protein of length N, we first translate the structure to match its center of geometry (α-carbon) with the origin of the coordinate system. We then rotate the protein around the origin to let the first residue be on the negative side of z-axis (principal component-based

orienting was also explored as in **Training and Decoding Strategy**). We denote the resulting residue coordinates as $c_1, c_2, ..., c_N$. The secondary structure label to each residue is assigned based on their SSE assignment (Kabsch & Sander, 1983) in Protein Data Bank (Berman et al., 2000). We consider 4 types of secondary structure labels: helix, beta strand, loop and bend/turn. In order to consider the distribution of different secondary structure labels in the 3D space, we discretize the 3D space into voxels. Due to the scale-free definition of a protein fold, we rescale the original structure, so that it fits into a fixed-size cubic box. Based on the distribution of sizes of single-chain, single-domain proteins in the CATH database (Sillitoe et al., 2019), we choose a $40\text{Å} \times 40\text{Å} \times 40\text{Å}$ box with each voxel of size $2\text{Å} \times 2\text{Å} \times 2\text{Å}$. We denote the scaling ratio as $r \in \mathcal{R}^3$. For voxel i, we denote the coordinates of its center as v_i. We assume that the contribution of residues j to voxel i follows a Gaussian form:

$$y_{ij} = \exp(-\frac{||c_j \odot r - v_i||_2^2}{\sigma^2}) \cdot t_j, \tag{1}$$

where $t_j \in \{0, 1\}^4$ is the one-hot encoding of the secondary structure label of amino acid j. The standard deviation is chosen to be 2Å. We sum up all residues together to obtain the final features of the voxel i: $y_i = \sum_{j=1}^{N} y_{ij}$. The fold representation $y \in \mathcal{R}^{20 \times 20 \times 20 \times 4}$ is the 4D tensor of y_i over all $20 \times 20 \times 20$ voxels. This fold representation using 3D SSE densities better captures scale-free SSE topologies that define folds, while removing fold-irrelevant structure details. It results in sequence generation that explores the sequence space available to a specific fold more widely (as shown in experiments).

3.3. Fold2Seq with Joint Sequence–Fold Embedding

Model Architecture. In the training stage, our model consists of three major components: a sequence encoder: $h_s(\cdot)$, a fold encoder: $h_f(\cdot)$ and a sequence decoder: $p(x|h(\cdot))$, as shown in Fig. 2 (Left).

(i) **Sequence Encoder/Decoder.** Both sequence encoder and decoder are implemented using the vanilla transformer model and a vanilla sequence embedding module (learnable
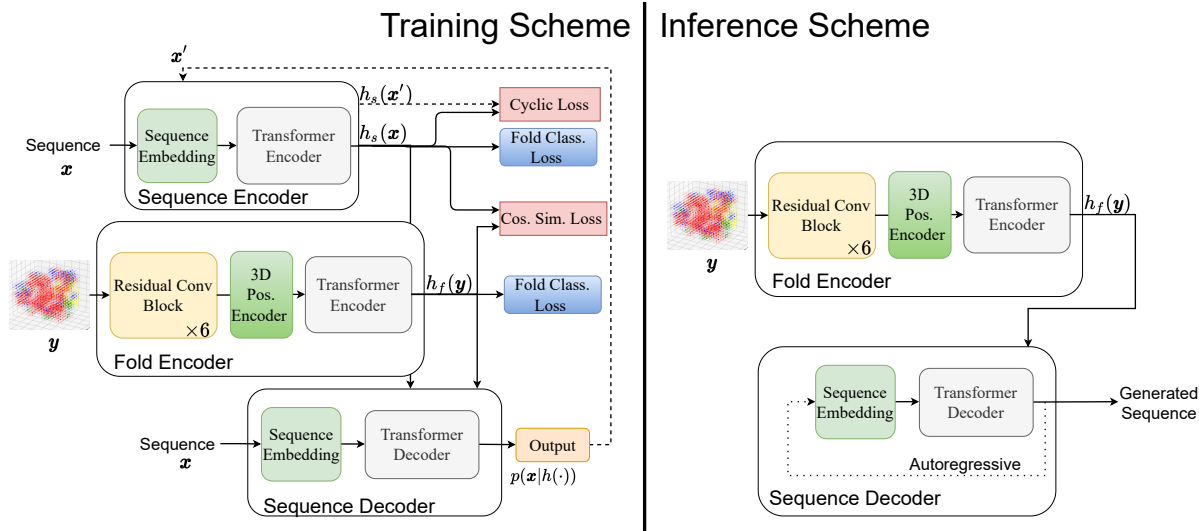
Figure 2: The architecture of the Fold2Seq model during the training and inference stages. (**Training Scheme**): During training, the model includes three major components: (top) Sequence Encoder, (middle) Fold Encoder and (bottom) Sequence Decoder. The dashed arrows represent the process for getting cyclic loss. (**Inference Scheme**): During the inference, the model only needs the fold encoder and the sequence decoder for conditionally decoding sequences.

lookup table + sinusoidal positional encoding), as described in Vaswani et al. (2017). All training sequences are padded to the maximum length N_s of 200, as 77% of single-domain sequence lengths in the CATH dataset are $\leqslant 200$.

(ii) **Fold Encoder.** A fold representation $\boldsymbol{y} \in \mathcal{R}^{20 \times 20 \times 20 \times 4}$ will go through a fold encoder, which contains 6 residual blocks followed by a 3D positional encoding. Each residual block has two 3D-convolutional layers ($3 \times 3 \times 3$) and batch normalization layers. The 6 residual blocks transform $\boldsymbol{y}$ to a tensor with the shape $5 \times 5 \times 5 \times d$, where d is the hidden dimension. The 3D positional encoding is a simple 3D extension of the sinusoidal encoding described in the vanilla transformer model, as shown in the Supporting Information (**SI**) **Sec. 1**. After the positional encoding, the 4D tensor is flattened to be 2D with the shape $125 \times d$, as the input of a transformer encoder. The output of the transformer encoder, $h_f(\boldsymbol{y})$, is the latent fold representation of $\boldsymbol{y}$.

We propose a simple fold-to-sequence reconstruction loss based on the auto-encoder model: $\mathbf{RE}_f = p(\boldsymbol{x}|h_f(\boldsymbol{y}))$. However, as mentioned earlier, training based on $\mathbf{RE}_f$ alone suffers from the heterogeneity of $\boldsymbol{x}$ and $\boldsymbol{y}$. To overcome this challenge, we first encode the sequence $\boldsymbol{x}$ through the sequence encoder into the latent space as $h_s(\boldsymbol{x})$, which could be done through a simple sequence-to-sequence reconstruction loss: $\mathbf{RE}_s = p(\boldsymbol{x}|h_s(\boldsymbol{x}))$. We then learn a joint latent space between $h_f(\boldsymbol{y})$ and $h_s(\boldsymbol{x})$ through a novel sequence-fold embedding learning framework with additional losses detailed below.

Joint Embedding Learning. Typically, learning a joint embedding across two domains needs two intra-domain

losses and one cross-domain loss (Chen et al., 2018). An intra-domain loss forces two semantically similar samples from the same domain to be close to each other in the latent space, while a cross-domain loss forces two semantically similar samples in different domains to be closer.

In our case, the meaning of 'semantically similar' is that the proteins should have the same fold(s). Therefore, we consider a supervised learning task for learning intra-domain similarity: fold classification. Specifically, the outputs of both encoders: $h_f(\boldsymbol{y}) \in \mathcal{R}^{l_f \times d}$ and $h_s(\boldsymbol{x}) \in \mathcal{R}^{l_s \times d}$ will be averaged along l_f and l_s dimensions, followed by a MLP+softmax layer to perform fold classification (shown as two blue blocks in Fig. 2), where l_s and l_f are the length of the sequence and the fold, respectively. The parameters of the two MLP layers are shared. The category labels follow the fold (topology) level of hierarchical protein structure classification in CATH4.2 dataset (Sillitoe et al., 2019) (see Section 3.4). As a result, we propose the following two intra-domain losses: $\mathbf{FC}_f$ and $\mathbf{FC}_s$, i.e. the cross entropy losses of fold classification from $h_f(\boldsymbol{y})$ and $h_s(\boldsymbol{x})$ respectively. The benefits of these two classification tasks are as follows: First, it will force the fold encoder to learn the fold representation. Second, as we perform the same supervised learning task on the latent vectors from two domains, it will not only learn the intra-domain similarity, but also cross-domain similarity. In contrast, without explicit cross-domain learning, the two latent vectors $h_f(\boldsymbol{y})$, $h_s(\boldsymbol{x})$ could still have minimal alignment between them.

In the transformer decoder, each element in the **non**-self attention matrix is calculated by the cosine similarity between the latent vectors from the encoder and the decoder,

respectively. Inspired by this observation, we maximize the cosine similarity (shown as the 'Cosine Similarity' in Fig. 2) between $h_f(\boldsymbol{y}) \in \mathcal{R}^{l_f \times d}$ and $h_s(\boldsymbol{x}) \in \mathcal{R}^{l_s \times d}$ as the cross-domain loss. We first calculate the matrix-product between $h_f(\boldsymbol{y})$ and $h_s(\boldsymbol{x})$ as $\boldsymbol{Q} = h_f(\boldsymbol{y}) \cdot h_s(\boldsymbol{x})^T, \boldsymbol{Q} \in \mathcal{R}^{l_f \times l_s}$. The ith row in $\boldsymbol{Q}$ represents the similarity between ith position in the fold and every position of the sequence. We would like to find the best-matching sequence piece with each position in the fold. To achieve this, the similarity matrix $\boldsymbol{Q}$ first goes through a row-wise average pooling with kernel size k, followed by the row-wise $\max$ operation:

$$\boldsymbol{q} = \max_{row}(\text{AvgPool}_{row}^k(\boldsymbol{Q})), \boldsymbol{q} \in \mathcal{R}^{l_f \times 1}, \qquad (2)$$

where row indicates row-wise operation. We choose $k = 3$, *i.e.* the scores of every 3 contiguous positions in the sequence will be averaged. We finally average over all positions in the fold to get the final similarity score: $\mathbf{CS} = \text{mean}(\boldsymbol{q})$.

Besides the cosine similarity loss, inspired by the earlier CycleGAN work (Zhu et al., 2017), we add a cyclic loss (shown as the red block of "Cyclic Loss" in Fig. 2.) to be another term of our cross-domain loss. Specifically, we take the argmax of the output of fold-to-sequence model: $\boldsymbol{x}' = \arg\max p(\boldsymbol{x}|h_f(\boldsymbol{y}))$, and send it back to the sequence encoder for generating the cyclic-seq latent state: $h_s(\boldsymbol{x}')$ (shown as the dashed line in Fig. 2). This cyclic-seq latent state will compare with the native seq latent state $h_s(\boldsymbol{x})$ through the square of the L2 distance:

$$\mathbf{CY} = ||h_s(\boldsymbol{x}') - h_s(\boldsymbol{x})||_2^2 \qquad (3)$$

To summarize, the complete loss objective is the following:

$$L = \lambda_1\mathbf{RE}_f + \lambda_2\mathbf{RE}_s + \lambda_3\mathbf{FC}_f + \lambda_4\mathbf{FC}_s + \lambda_5(\mathbf{CY} - \mathbf{CS}), \qquad (4)$$

where λ_1 through λ_5 are the hyperparameters for controlling the relative importance among these losses.

Training and Decoding Strategy. During experiments we found that, if the sequence encoder and the fold encoder were trained together, the fold encoder had little parameter improvement while the sequence encoder dominated the training. To overcome this issue, we consider a two-stage training strategy. In the first stage, we train the sequence-to-sequence model regularized by the sequence intra-domain loss: $L_1 = \lambda_2\mathbf{RE}_s + \lambda_4\mathbf{FC}_s$. After the first stage is finished, we start the second training stage. We train the fold-to-sequence model regularized by the fold intra-domain loss and the cross-domain loss while keeping the sequence encoder frozen: $L_2 = \lambda_1\mathbf{RE}_f + \lambda_3\mathbf{FC}_f + \lambda_5(\mathbf{CY} - \mathbf{CS})$. The comparison between the one-stage training and two-stage training strategies are described in details in **SI Sec.** 2.

We implement our model in Pytorch (Paszke et al., 2019). Each transformer block has 4 layers and $d = 256$ latent dimensions. In order to increase the robustness of our model

for rotated structures, we augment our training data by right-hand rotating the each structure by $90°$, $180°$ and $270°$ along each axis (x,y,z). As a result, we augment our training data by $3 \times 3 = 9$ folds. While orienting proteins along principal axes is better using global shapes, we find neither orientations along principal axes nor denser augmentations ($45°$) empirically boosted the model performance (See Result in Sec. 4). The learning rate schedule follows the original transformer paper (Vaswani et al., 2017). We use the exponential decay (Blundell et al., 2015) for $\lambda_5 = 1/2^{\#\text{epoch}-e}$ in the loss function, while λ_1 through λ_4 and e are tuned based on the validation set, resulting in $\lambda_1 = 1.0, \lambda_2 = 1.0, \lambda_3 = 0.02, \lambda_4 = 1.0, e = 3$. We train our model on 2 Tesla K80 GPUs, with batch size 128. In every training stage we train up to 200 epochs with an early stopping strategy based on the validation loss.

During inference, one only needs the fold encoder and the sequence decoder for conditional sequence generation (Fig. 2 (Right)). Top-k sampling strategy (Fan et al., 2018) is used for sequence generation, where k is tuned to be 5 based on the validation set.

3.4. Benchmark Datasets

We used protein structure data from CATH 4.2 (Sillitoe et al., 2019) filtered by 100% sequence identity. We remove proteins that (1) are multi-chain or non-contiguous in sequence; (2) contain other than 20 natural amino acids; or (3) have length longer than 200. We randomly split the dataset at the fold level into 95%, 2.5%, 2.5% as dataset (a), (b) and (c), respectively, which means that the three datasets have non-overlapping folds. We further randomly split the dataset (a) at the structure level into 95%, 2.5% and 2.5% as dataset (a1), (a2) and (a3), respectively. Datasets (a1), (a2), and (a3) have overlapping folds. We use dataset (a1) as the training set, (b)+(a2) as the validation set, (a3) as the In-Distribution (ID) test set and (c) as the Out-of-Distribution (OD) test set. The folds of the ID test set overlap with the training set, whereas the folds of the OD test set do not. Statistics of these datasets are presented in **SI Sec. 3**.

To quantitatively measure their difficulty levels, we calculate the **a**veraged **m**aximum **s**equence **i**dentity (**amsi**) between a given test set $\boldsymbol{T}$ and the training set as: $\text{amsi}_{\boldsymbol{T}} = \frac{1}{|D_{\boldsymbol{T}}|}\sum_{j \in D_{\boldsymbol{T}}} \max_{k \in D_{\text{train}}}(\text{SIM}(\boldsymbol{x}_j, \boldsymbol{x}_k))$, where D_{train} and $D_{\boldsymbol{T}}$ are the training and test ($\boldsymbol{T}$) set, respectively; $\text{SIM}(\boldsymbol{x}_j, \boldsymbol{x}_k)$ is the sequence identity (See **SI Sec. 4**) between sequence $\boldsymbol{x}_j$ and $\boldsymbol{x}_k$. We found $\text{amsi}_{\text{ID}} = 36.3\%$ and $\text{amsi}_{\text{OD}} = 16.3\%$, showing that the OD test set represents a much more difficult generalization task compared to the ID.

3.5. Assessment Metrics

Ideally, the most appropriate and rigorous criteria for evaluating fold-based protein design methods is to check the

consistency between the structures of the generated sequences and the desired fold. However, as protein structure prediction is very computationally expensive, and similar sequences usually indicate similar folds or structures, many earlier structure-based methods (Ingraham et al., 2019; Madani et al., 2020) report performance in the sequence domain. Here, considering that a fold is comprised of multiple structures, we define four fold-level metrics that is able to assess the quality of the designed sequences for a desired fold. For a test set $D_{\mathbf{T}}$, we use $i \in D_{\mathbf{T}}$ to represent a fold in $D_{\mathbf{T}}$ and $j \in D_{\mathbf{T}}$ (or $k \in D_{\mathbf{T}}$) to represent a protein in $D_{\mathbf{T}}$.

(Fold-level) Per-residue perplexity. Based on (Ingraham et al., 2019; Madani et al., 2020), the structure-level per-residue PerPLexity (ppl) for test $D_{\mathbf{T}}$ is defined as: $ppl_{\text{structure}}(D_{\mathbf{T}}) = \exp(-\frac{1}{|D_{\mathbf{T}}|} \sum_{j \in D_{\mathbf{T}}} \frac{1}{L_j} \log p(\boldsymbol{x}_j | \boldsymbol{y}_j))$, where L_j is the length of sequence j. Here we consider per-residue perplexity for fold i:

$$ppl_{\text{fold}}(i) = \exp(-\frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \max_{k \in \mathcal{S}_i}(\frac{1}{L_k} \log p(\boldsymbol{x}_k | \boldsymbol{y}_j)), \quad (5)$$

where $\mathcal{S}_i$ is set of structures in fold i. We compute the mean and standard deviation of $ppl_{\text{fold}}(i)$ over all folds in $D_{\mathbf{T}}$.

(Fold-level) Sequence Recovery. We define the set of the generated sequences (structures) conditioned on structure j as $\mathcal{G}_j$. In structure-based design, we usually define the Sequence Recovery rate (sr) for $\boldsymbol{y}_j$ as $sr_{\text{structure}}(j) = \frac{1}{|\mathcal{G}_j|} \sum_{g \in \mathcal{G}_j} \text{SIM}(\boldsymbol{x}_g, \boldsymbol{x}_j)$. Here we consider the fold-level sequence recovery rate for fold i:

$$sr_{\text{fold}}(i) = \frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \frac{1}{|\mathcal{G}_j|} \sum_{g \in \mathcal{G}_j} \max_{k \in \mathcal{S}_i}\{\text{SIM}(\boldsymbol{x}_g, \boldsymbol{x}_k)\}. \quad (6)$$

(Fold-level) Coverage (Diversity). We also measure how the generated sequences from a single (or few) representative structure(s) could capture the diversity of sequences (thus of structures and functions) within a fold. To do so, for fold i, we randomly pick **one** structure k from $\mathcal{S}_i$ as the representative. We then measure how many sequences within that fold are captured by the generated sequences conditioned on the representative. As a result, we define the COVerage (cov) for fold i as:

$$cov_{\text{fold}}(i) = \frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \mathbb{1}(\max_{g \in \mathcal{G}_k}\{\text{SIM}(\boldsymbol{x}_g, \boldsymbol{x}_j)\} \geqslant 30\%). \quad (7)$$

We use the rule of thumb: two sequences likely belong to the same fold if their identity is above 30% (Rost, 1999).

(Fold-level) Structure Recovery. We last assess design accuracy directly in the structure domain. In structure-based design, similar to the sequence recovery, the sTructure Recovery (tr) rate is defined as: $tr_{\text{structure}}(j) =$

$\frac{1}{|\mathcal{G}_j|} \sum_{g \in \mathcal{G}_j} \text{TM}(\boldsymbol{y}_g, \boldsymbol{y}_j)$, where $\text{TM}(\boldsymbol{y}_g, \boldsymbol{y}_k)$ is the TM-score (Zhang & Skolnick, 2004) between structures $\boldsymbol{y}_g$ and $\boldsymbol{y}_k$. Here we extend it to fold-level structure recovery:

$$tr_{\text{fold}}(i) = \frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \frac{1}{|\mathcal{G}_j|} \sum_{g \in \mathcal{G}_j} \max_{k \in \mathcal{S}_i}\{\text{TM}(\boldsymbol{y}_g, \boldsymbol{y}_k)\}. \quad (8)$$

We used the iTasser Suite (Yang et al., 2015), one of the state-of-the-art protein structure prediction software, to predict the structure of the designed sequences. For all metrics in the sequence domain, we have set $|\mathcal{G}_j| = 100$ for every $j \in D_{\mathbf{T}}$. However, as iTasser usually takes at least one day for predicting the structure of a single protein, for tr_{fold} in the structure domain, we use $|\mathcal{G}_j| = 1$ for every $j \in D_{\mathbf{T}}$. We also include the performances of different methods based on the structure-level metrics in **SI Sec. 5**.

3.6. Baseline Methods

Data-driven. We consider two data-driven *fold-based* methods that design sequences conditioned on a desired fold: cVAE (Greener et al., 2018) and gcWGAN (Karimi et al., 2020). We also consider a recent *structure-based* method, Graph_trans (Ingraham et al., 2019), that uses graph specification on the backbone structure as input and has shown to outperform earlier structure-based methods in terms of the structure-level metrics. We used Graph_trans conditioned on the flexible backbone for comparison. **Physics-based.** We then consider the state-of-the-art principle-driven method, RosettaDesign[1] (Huang et al., 2011).

4. Experiments on Benchmark Test Sets.

Perplexity and Sequence Recovery Comparison. We first compare ppl_{fold} of Fold2Seq with those of the baseline methods (except RosettaDesign, as ppl_{fold} is not applicable to it). For reference, we also show the per-residue perplexity under the uniform distribution and the frequencies through all natural sequences in UniRef50 (Suzek et al., 2015). We do not report standard deviation on these perplexities as they are unconditional distributions. Performances on two test sets are summarized in Table 1a, showing that Fold2Seq has the smallest ppl_{fold} on ID test set and the second smallest on OD test set. We have also tested different data augmentation strategies including orientations along principal axes and denser augmentations ($45°$). Neither strategy significantly boosted the model performance for the OD set (ppl_{fold}: 12.2 (2.7) and 12.0 (2.5), respectively).

Next, we compare different methods for recovering the native sequences within a desired fold. Also, for comparison, we calculate the expected similarity between two random

---

[1]RosettaDesign uses MCMC sampling and energy calculation to search for best sequences. The input to RosettaDesign consists of the backbone of the native structure and the SSE of each residue.

Table 1: Performance of different methods assessed by (a) Avg. ppl_fold (std. dev.) and (b) Avg. sr_fold (std. dev.) (%).

(a)

Model	ID Test	OD Test
Uniform	20.0	20.0
Natural	18.0	18.0
cVAE	13.2 (2.2)	15.2 (2.3)
gcWGAN	12.3 (2.3)	14.3 (2.5)
Graph_trans	9.6 (2.9)	**11.5 (3.3)**
Fold2Seq	**9.0 (5.3)**	12.0 (2.4)

(b)

Model	ID Test	OD Test
Random across two folds	12.8 (7.9)	12.8 (7.9)
cVAE	18.2 (6.7)	17.3 (5.2)
gcWGAN	20.6 (5.4)	19.2 (3.7)
RosettaDesign	22.1 (5.7)	22.3 (3.5)
Graph_trans	**28.8 (11.3)**	**27.1 (4.0)**
Fold2Seq	27.2 (6.3)	25.2 (3.2)
Random within same fold	39.1 (9.4)	39.1 (9.4)

sequences in our whole dataset belonging to two different folds and belonging to the same folds. The results are summarized in Table 1b. Overall, Graph_trans and Fold2Seq outperform other methods by a big margin, while Graph_trans shows slightly better performance than Fold2Seq. This is because Graph_trans is the only baseline that utilizes more high-resolution structural information beyond the fold level as the input. However, a structure-based method may not capture the similarity and diversity within the fold space. To highlight this point, we use t-SNE to visualize the fold embeddings h after the fold encoder for the proteins in the OD test sets. The results in **SI Sec. 8** evidently show that the embeddings of same-fold proteins from Graph_trans are less clustered than those from Fold2Seq.

Coverage (Diversity). Coverage, as defined in Eq 7, is shown in Table 2a. We split the folds within the testsets based on the number of sequences within each fold ($|\mathcal{S}_i|$) using a cutoff of 3. Overall, Fold2Seq shows better coverage, compared to other baselines. In most cases, coverages on more diverse folds ($|\mathcal{S}_i| > 3$) have smaller standard deviations due to large $|\mathcal{S}_i|$ in the denominator of Eq. 7. We then directly compare Fold2Seq with Graph_trans by counting the number of folds for which Fold2Seq yields better $cov_\text{fold}(i)$. As shown in Table 2b, Fold2Seq provides better coverage in 68%-88% folds, implying that the proposed method can better capture the diversity within a fold compared to Graph_trans.

Moreover, we compare with an alternative version of Graph_trans: Graph_trans_all, which conditions on each structure within a fold and then combines the sequences generated over all conditions (instead of one) for calculating the coverage. Though such an approach treats structure inputs separately and do not target what makes diverse structures

common to a fold or distinguished across folds (evident in visualization of learned embeddings in **SI Sec. 8**). Table 2a shows that Fold2Seq outperforms Graph_trans_all in most cases except in the OD test set with $|\mathcal{S}_i| > 3$.

Table 2: (a) Avg. cov_fold (std. dev. in %). (b) Fold2Seq(f) and Graph_trans(g) head-to-head coverage comparison.

(a)

	ID Test		OD Test	
Subset	$\|\mathcal{S}_i\| \leqslant 3$	$\|\mathcal{S}_i\| > 3$	$\|\mathcal{S}_i\| \leqslant 3$	$\|\mathcal{S}_i\| > 3$
cVAE	16.2 (17.3)	13.3 (16.1)	15.2 (16.3)	11.3 (12.4)
gcWGAN	18.9 (15.3)	20.5 (21.2)	17.3 (13.4)	15.3 (12.8)
Graph_trans	19.4 (28.9)	24.1 (25.1)	26.9 (32.5)	20.2 (19.8)
Graph_trans_all	28.9 (32.3)	25.3 (30.1)	30.2 (25.2)	**21.3 (23.7)**
RosettaDesign	20.3 (17.3)	17.3 (16.2)	21.2 (20.3)	17.5 (18.9)
Fold2Seq	**32.9 (33.5)**	**28.9 (27.8)**	**34.3 (38.3)**	20.7 (17.7)

(b)

	ID Test		OD Test	
Subset	$\|\mathcal{S}_i\| \leqslant 3$	$\|\mathcal{S}_i\| > 3$	$\|\mathcal{S}_i\| \leqslant 3$	$\|\mathcal{S}_i\| > 3$
$\#cov_\text{fold}^\text{f}(i) > cov_\text{fold}^\text{g}(i)$	104	53	13	8
Total #folds	118	78	18	10
Ratio	0.88	0.68	0.72	0.80

Structural Recovery Comparison with RosettaDesign. Besides sequence-domain assessments, we examine if the structure of a Fold2Seq-generated sequence is of the same or similar folds to the native structure. Due to the associated computational expense, we limit structure predictions to proteins designed by Fold2Seq and RosettaDesign. We first compare the distributions of $ts_\text{fold}(i)$ on two test sets (ID and OD); results are shown in Fig. 3(a). Fold2Seq shows significant improvement against RosettaDesign. The performance of Fold2Seq on ID test set is better than that on OD, thus matching their expected difficulty levels. RosettaDesign performs similarly on both sets due to its physics-based nature that does not rely on learning from a training set.

To quantitatively measure the performance difference between the two methods, we define $\Delta ts_\text{fold}(i) = ts_\text{fold}^\textbf{Fold2Seq}(i) - ts_\text{fold}^\textbf{Rosetta}(i)$ and perform one-sided one-sampled t-test over Δts_fold, with the null hypothesis as "$\Delta ts_\text{fold} \leqslant 0.0$" on two test sets. The resulting **P-value**$_\text{ID} = 1.58\text{E} - 23$ and **P-value**$_\text{OD} = 0.00012$ demonstrate that, overall, Fold2Seq can generate more reliable structures compared to RosettaDesign. The two distributions over Δts_fold are shown in Fig 3(b-c). We also randomly pick some designed structures within the fold i with $\Delta ts_\text{fold}(i) > 0.0$ and $\Delta ts_\text{fold}(i) < 0.0$, and visualize them in Fig. S4 and Fig. S5 in **SI Sec. 7**, respectively.

The computational efficiency in terms of inference is shown for Fold2Seq and RosettaDesign in Fig. 3(d) . Compared to RosettaDesign on CPU, Fold2Seq on CPU and that on GPU are almost 100 times and 5000 times faster, respectively.

Generalizability Analysis. For each fold in the test sets, we calculate the maximum sequence identity (MSI) between
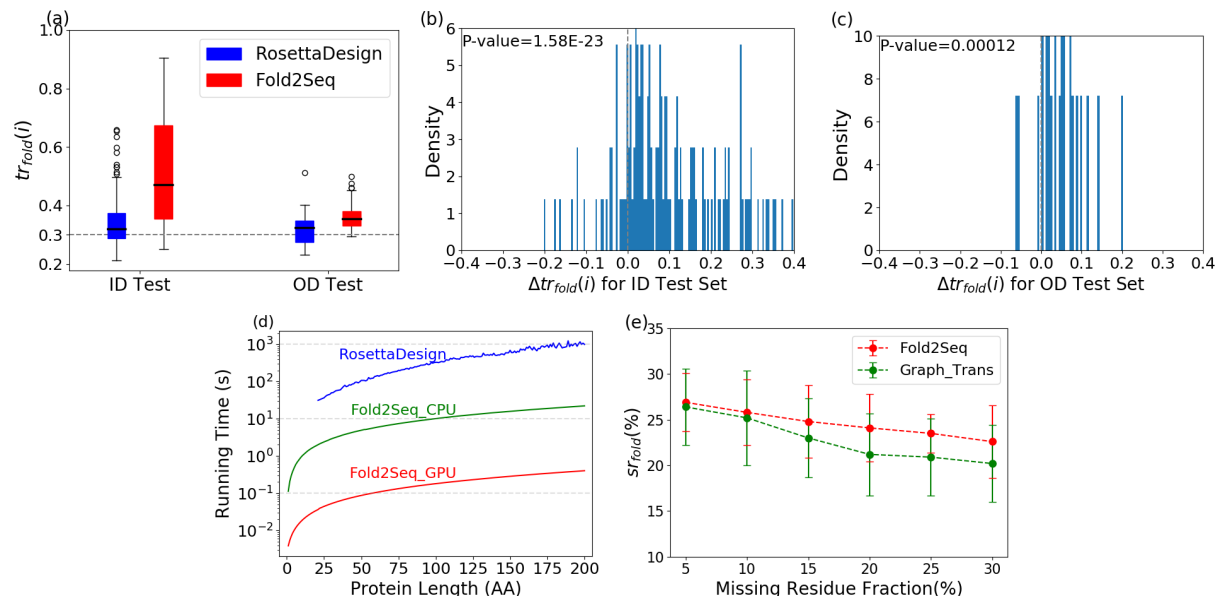
Figure 3: (a). $tr_{\text{fold}}(i)$ distributions of RosettaDesign and Fold2Seq. (b, c). The distributions of $\Delta tr_{\text{fold}}(i)$ for ID test set and OD test set, respectively. (d). Run time of Fold2Seq and RosettaDesign for generating one protein sequence: CPU: Intel Xeon E5-2680 v4 2.40GHz, GPU: Nvidia Tesla K80. (e). Avg. $sr_{\text{fold}}(i)$ for the OD test set with a continuous stretch of missing residues.

a randomly selected sequence from that fold and all folds in the training set (one random sequence per fold). We split all folds in the test set into several bins. The performances of ppl_{fold}, sr_{fold} and cov_{fold} over bins are shown in **SI** Fig. S3. In most cases, as the MSI increases, all methods have better performances in all three metrics except RosettaDesign which does not need a training set. For the low MSI bins that demand generalizability, Fold2Seq is the best performer in ppl_{fold} on the ID test set and in cov_{fold} on both test sets, as well as the second best (next to Graph_trans) in ppl_{fold} on the OD test set and in sr_{fold} on both test sets.

Ablation Study. To rigorously delineate the contributions of each algorithmic innovation, we perform an ablation study (detailed in **SI Sec. 9**). The performance on the two test sets in terms of averaged sr_{fold} is summarized in Table 3a. **Key** observations are: (i) 'String' to 'voxel' change and addition of 2 **FC** losses provide the largest performance gain (2-3%). (ii) Use of transformer and the cyclic loss improves performance by around 2%. (iii) In contrast, the improvement due to the addition of **RE**$_s$ and **CS** is minor. (iv) Further, the inclusion of the two **FC** losses as the intra-domain loss is crucial for joint embedding learning. By calculating the averaged pairwise L2 distance among the hidden fold vectors, $h_f(\boldsymbol{y})$, for proteins in the OD test set, we found that such distance is smaller with FC losses (3.25) than without FC losses (5.35), which echoes our rationales of proposing fold-classification losses in the Method section. In summary, our novel design of the 3D voxel representation and the joint embedding learning framework, which

Table 3

(a) Avg. sr_{fold} (std. dev.) (%) for variants in ablation study.

Model	ID Test	OD Test
cVAE	18.2 (6.7)	17.3 (5.2)
Trans_string_**RE**$_f$	20.0 (8.31)	19.2 (3.45)
Trans_voxel_**RE**$_f$	22.5 (7.34)	21.3 (3.33)
+**RE**$_s$+**CS**	22.8 (8.01)	21.9 (2.34)
+2**FC**	25.6 (6.34)	23.7 (2.34)
+**CY** (Fold2Seq)	**27.2 (6.3)**	**25.2 (3.2)**

(b) Avg. $sr_{\text{structure}}$ (std. dev.) (%) for low resolution structures, and Avg. sr_{fold} and cov_{fold} (std. dev.) (%) for NMR ensembles.

Model	Low_res Set	
Graph_trans	19.9 (4.8)	
RosettaDesign	17.2 (6.3)	
Fold2Seq	**21.2 (3.1)**	
sr_{fold} for NMR	ID Test	OD Test
Fold2Seq$_{\text{single}}$	24.1 (3.9)	22.2 (3.8)
Fold2Seq$_{\text{average}}$	25.2 (3.5)	24.1 (4.2)
cov_{fold} for NMR	ID Test	OD Test
Graph_trans$_{\text{all}}$	19.5 (26.3)	17.5 (28.5)
Fold2Seq$_{\text{single}}$	24.1 (3.9)	22.2 (3.8)
Fold2Seq$_{\text{average}}$	**25.2 (3.5)**	**24.1 (4.2)**

includes intra-domain and cyclic losses, leads to significant performance improvement.

5. Experiments on Real-world Challenges.

To further explore the practical utility of our model, we perform three real-world challenging design tasks condi-

tioned on: (1) Low-resolution structures; (2) Structures with missing residues; and (3) NMR ensembles, representing low-quality, incomplete, and ambiguous data, respectively.

Low-Resolution Structures. We first create the low-resolution structure dataset from Protein Data Bank, which contains 164 single-chain proteins with low resolutions ranging from 6Å to 12Å. This set has maximum sequence identity (MSI) below 30% compared to the training set. We compare Fold2Seq's performance on this set with those of Graph_trans and RosettaDesign. Since the fold information is not available for these low-resolution structures, we report structure-level sequence recovery ($sr_{\text{structure}}$) in Table 3b, showing that Fold2Seq outperforms other baselines. As Fold2Seq uses the high-level fold information (by re-scaling the structure, discretizing the space, and smoothing the spatial secondary structure element information by neighborhood averaging), the model's performance is less sensitive compared to RosettaDesign or Graph_trans, when test structures are of lower resolution. To further solidify the results, we randomly pick three Fold2Seq's designed sequences on three different proteins respectively and recover their structures through iTasser Suite. As a result, we receive $tr_{\text{structure}} : 0.39, 0.46, 0.33$ on protein 2W6G_G, 5BW9_L and 5UJ8_H, respectively, indicating structural similarity with the target structure.

Structures with Missing Residues. We next perform the design task where the input structures have missing residues. In order to mimic the real-world scenario, for every protein in our OD test set, we select a stretch of residues at random starting positions with length p, for which 1D and 3D information was removed. We compared Fold2Seq with Graph_trans at $p = \{5\%, 10\%, 15\%, 20\%, 25\%, 30\%\}$, as shown in Fig. 3(e). When p is small, the performance of Fold2Seq is on par with Graph_trans. As p increases, Fold2Seq outperforms Graph_trans with a consistent margin. We also perform one-sided, two-sample t-tests with null hypothesis: sr_{fold} of Graph_trans is larger than that of Fold2Seq and obtain P-value of 0.028 (at 10% missing rate) or <1E-3 (at higher missing rates). This shows that Fold2seq is less sensitive to the availability of complete and detailed backbone structure information.

NMR Structural Ensemble. We finally apply Fold2Seq to a structural ensemble of NMR structures. We filter the NMR structures from our two test sets and obtain 57 proteins in 30 folds from the ID set and 30 proteins in 10 folds from the OD set. On average each protein has around 20 structures. Handling NMR ensembles using Fold2Seq is straightforward, when compared to Graph_trans and RosettaDesign: after we obtain the voxel-based features through Eq 1 for each model (structure) within one NMR ensemble, we simply average them across all models. The sequence recovery results of Fold2Seq for NMR ensembles are shown in Table 3b, along with a single structure baseline. Results show

that Fold2Seq performs better on both ID and OD proteins, when ensemble structure information is available. This is consistent with our hypothesis that our fold representation better captures the structural variations present within a single fold. Moreover, we compare the coverage performance of Fold2Seq against multiple Graph_trans which collectively uses the ensemble of all models within a NMR structure, and all NMR structures within a fold. As shown in Table 3b, Fold2Seq designs using single or averaged SSE densities achieved higher coverage than Graph_trans did using all structures, which shows that Fold2Seq has better efficiency and scalability for inverse fold design compared to structure-based methods with diverse structure inputs.

6. Conclusion and Future Work

In this paper, we design a novel transformer-based model to learn a fold representation from scale-free and coarse topological features extracted from 3D voxels of secondary structure elements within and across folds and use those as conditional inputs to design protein sequences. In order to mitigate the heterogeneity between the sequence domain and the fold domain, we learn the joint sequence–fold representation through novel intra-domain and cross-domain losses. On benchmark datasets containing single, high-resolution, complete input structures, Fold2Seq performs better or similarly, compared to the existing neural net models and the state-of-the-art principle-driven RosettaDesign method, in terms of perplexity, sequence recovery rate, coverage and structural recovery. Ablation study shows that this superior performance can be directly attributed to our novel algorithmic innovations, including the fold representation, joint sequence-fold embedding, and various losses. Moreover, we demonstrate the unique practical utility of Fold2Seq compared to structure-based neural net models in a set of real-world design tasks with challenging conditional inputs such as low resolution structures, structures with region of missing residues, and NMR structural ensembles.

Future work will focus on upgrading fold embedding from convolutional neural networks to advanced architectures such as certain SE(3)-equivariant ones, learning representations in a continuous rather than a discrete fold space, and designing multi-domain and multi-chain proteins.

Acknowledgements

References

Arandjelovic, R. and Zisserman, A. Objects that sound. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 435–451, 2018.

Basanta, B., Bick, M. J., Bera, A. K., Norn, C., Chow, C. M., Carter, L. P., Goreshnik, I., Dimaio, F., and Baker, D. An enumerative algorithm for de novo design of proteins with diverse pocket structures. *Proceedings of the National Academy of Sciences*, 117(36):22135–22145, 2020.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. The protein data bank. *Nucleic acids research*, 28(1): 235–242, 2000.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Boutemy, L. S., King, S. R., Win, J., Hughes, R. K., Clarke, T. A., Blumenschein, T. M., Kamoun, S., and Banfield, M. J. Structures of phytophthora rxlr effector proteins: a conserved but adaptable fold underpins functional diversity. *Journal of Biological Chemistry*, 286(41):35834–35842, 2011.

Cao, Y. and Shen, Y. TALE: Transformer-based protein function Annotation with joint sequence–Label Embedding. *Bioinformatics*, 03 2021. ISSN 1367-4803. doi: 10.1093/bioinformatics/btab198. btab198.

Chandra, N. R., Prabu, M., Suguna, K., and Vijayan, M. Structural similarity and functional diversity in proteins containing the legume lectin fold. *Protein Engineering*, 14(11):857–866, 2001.

Chen, K., Choy, C. B., Savva, M., Chang, A. X., Funkhouser, T., and Savarese, S. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian Conference on Computer Vision*, pp. 100–116. Springer, 2018.

Chen, S., Sun, Z., Lin, L., Liu, Z., Liu, X., Chong, Y., Lu, Y., Zhao, H., and Yang, Y. To improve protein sequence profile prediction through image captioning on pairwise residue distance map. *Journal of chemical information and modeling*, 60(1):391–399, 2019.

Das, P., Wadhawan, K., Chang, O., Sercu, T., Santos, C. D., Riemer, M., Chenthamarakshan, V., Padhi, I., and Mojsilovic, A. Pepcvae: Semi-supervised targeted design of antimicrobial peptide sequences. *arXiv preprint arXiv:1810.07743*, 2018.

David, L., Thakkar, A., Mercado, R., and Engkvist, O. Molecular representations in ai-driven drug discovery: a review and practical guide. *Journal of Cheminformatics*, 12(1):1–22, 2020.

Dognin, P., Melnyk, I., Mroueh, Y., Ross, J., and Sercu, T. Adversarial semantic alignment for improved image captions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10463–10471, 2019.

Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

Greener, J. G., Moffat, L., and Jones, D. T. Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific reports*, 8(1):1–12, 2018.

Gu, J., Cai, J., Joty, S. R., Niu, L., and Wang, G. Look, imagine and match: Improving textual-visual cross-modal retrieval with generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7181–7189, 2018.

Hou, J., Sims, G. E., Zhang, C., and Kim, S.-H. A global representation of the protein fold space. *Proceedings of the National Academy of Sciences*, 100(5):2386–2390, 2003.

Huang, P.-S., Ban, Y.-E. A., Richter, F., Andre, I., Vernon, R., Schief, W. R., and Baker, D. Rosettaremodel: a generalized framework for flexible backbone protein design. *PloS one*, 6(8):e24109, 2011.

Huang, P.-S., Boyken, S. E., and Baker, D. The coming of age of de novo protein design. *Nature*, 537(7620): 320–327, 2016.

Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. Generative models for graph-based protein design. In *Advances in Neural Information Processing Systems*, pp. 15820–15831, 2019.

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

Karimi, M., Zhu, S., Cao, Y., and Shen, Y. De novo protein design for novel folds using guided conditional wasserstein generative adversarial networks. *Journal of Chemical Information and Modeling*, 60(12):5667–5681, 2020.

Koga, N., Tatsumi-Koga, R., Liu, G., Xiao, R., Acton, T. B., Montelione, G. T., and Baker, D. Principles for designing ideal protein structures. *Nature*, 491(7423):222–227, 2012.

Kraemer-Pecore, C. M., Wollacott, A. M., and Desjarlais, J. R. Computational protein design. *Current Opinion in Chemical Biology*, 5(6):690–695, 2001.

Kuhlman, B., Dantas, G., Ireton, G. C., Varani, G., Stoddard, B. L., and Baker, D. Design of a novel globular protein fold with atomic-level accuracy. *science*, 302(5649):1364–1368, 2003.

Madani, A., McCann, B., Naik, N., Keskar, N. S., Anand, N., Eguchi, R. R., Huang, P.-S., and Socher, R. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

Murzin, A. G., Brenner, S. E., Hubbard, T., Chothia, C., et al. Scop: a structural classification of proteins database for the investigation of sequences and structures. *Journal of molecular biology*, 247(4):536–540, 1995.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. Y. Multimodal deep learning. In *ICML*, 2011.

O'Connell, J., Li, Z., Hanson, J., Heffernan, R., Lyons, J., Paliwal, K., Dehzangi, A., Yang, Y., and Zhou, Y. Spin2: Predicting sequence profiles from protein structures using deep neural networks. *Proteins: Structure, Function, and Bioinformatics*, 86(6):629–633, 2018.

Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B., and Thornton, J. M. Cath–a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pp. 8026–8037, 2019.

Peng, Y. and Qi, J. CM-GANs: Cross-modal generative adversarial networks for common representation learning. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 15(1):1–24, 2019.

Rost, B. Twilight zone of protein sequence alignments. *Protein engineering*, 12(2):85–94, 1999.

Sillitoe, I., Dawson, N., Lewis, T. E., Das, S., Lees, J. G., Ashford, P., Tolulope, A., Scholes, H. M., Senatorov, I., Bujan, A., et al. Cath: expanding the horizons of structure-based functional annotations for genome sequences. *Nucleic acids research*, 47(D1):D280–D284, 2019.

Socher, R., Ganjoo, M., Manning, C. D., and Ng, A. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*, pp. 935–943, 2013.

Strokach, A., Becerra, D., Corbi-Verge, C., Perez-Riba, A., and Kim, P. M. Fast and flexible design of novel proteins using graph neural networks. *bioRxiv*, pp. 868935, 2020.

Suzek, B. E., Wang, Y., Huang, H., McGarvey, P. B., Wu, C. H., and Consortium, U. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

Taylor, W. R. A 'periodic table' for protein structures. *Nature*, 416(6881):657–660, 2002.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., and Gamon, M. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1499–1509, 2015.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Wang, G., Li, C., Wang, W., Zhang, Y., Shen, D., Zhang, X., Henao, R., and Carin, L. Joint embedding of words and labels for text classification. *arXiv preprint arXiv:1805.04174*, 2018a.

Wang, J., Cao, H., Zhang, J. Z., and Qi, Y. Computational protein design with deep learning neural networks. *Scientific reports*, 8(1):1–9, 2018b.

Wang, K., He, R., Wang, W., Wang, L., and Tan, T. Learning coupled feature spaces for cross-modal matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2088–2095, 2013.

Woolfson, D. N., Bartlett, G. J., Burton, A. J., Heal, J. W., Niitsu, A., Thomson, A. R., and Wood, C. W. De novo protein design: how do we expand into the universe of possible protein structures? *Current opinion in structural biology*, 33:16–26, 2015.

Yang, J., Yan, R., Roy, A., Xu, D., Poisson, J., and Zhang, Y. The i-tasser suite: protein structure and function prediction. *Nature methods*, 12(1):7–8, 2015.

Zhang, Y. and Skolnick, J. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57(4):702–710, 2004.

Zhang, Z. and Saligrama, V. Zero-shot learning via joint latent similarity embedding. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6034–6042, 2016.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.