DISTILLING STRUCTURAL REPRESENTATIONS INTO PROTEIN SEQUENCE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein language (or sequence) models, like the popular ESM2, are now widely used tools for extracting evolution-based protein representations and have achieved significant success on core downstream biological tasks. A major open problem is how to obtain representations that best capture both the sequence evolutionary history and the atomic structural properties of proteins in general. We introduce ***I**mplicit **S**tructure **M**odel* (*ISM*), a sequence-only input model with structurally-enriched representations that outperforms state-of-the-art sequence models on several well-studied benchmarks including mutation stability assessment and structure prediction. Our key innovations are a microenvironment-based Autoencoder for generating structure tokens and a self-supervised training objective that distills these tokens into ESM2's pre-trained model. Notably, we make *ISM*'s structure-enriched weights easily accessible for any application using the ESM2 framework.

1 INTRODUCTION

Protein language models (pLMs) are versatile feature extractors with proven success across numerous downstream applications (Elnaggar et al., 2021; Brandes et al., 2022; Rives et al., 2019; Lin et al., 2022). Their accessibility has significantly democratized protein research, enabling biologists with limited computational resources or expertise to apply advanced machine learning techniques to their specific areas of study. The method's success comes from its exclusive use of sequences, bypassing costly, unreliable, or infeasible structure computations and sophisticated data-engineering pipelines.

The tradeoff is that pLMs are often lack structural context, and underperform (relative to structure-based models) on tasks that typically require structural insight (Su et al., 2023; Yang et al., 2023; Zhang et al., 2024; Gaujac et al., 2024; Frolova et al., 2024; Li et al., 2024). Longstanding biological research (Anfinsen, 1973) does suggest that the amino acid sequence is solely responsible for the folding of the structure. However, current state-of-the-art frameworks, such as AlphaFold, require the protein's evolutionary history as an additional input, and single-sequence frameworks, such as ESMfold, achieve subpar structure prediction performance. Building a *single-sequence* model that leads to rich structurally-informed representations remains a challenging open problem.

In this paper, we introduce ***I**mplicit **S**tructure **M**odel* (*ISM*), a sequence-only protein language model that is trained to *implicitly* capture structural information. Our key contribution is a novel self-supervised pre-training objective, *structure-tuning*, where the sequence model learns to distill features derived from structure-based models (see Figure 1). As a result, *ISM* outperforms sequence-only models and is competitive with pLM frameworks that *explicitly* take the protein structure as an additional input. On the CAMEO protein structure prediction benchmark, for example, *ISM* outperforms its ESM2 counterpart with a GDT-TS score of 0.67 versus 0.64 (see Table 1). For S669 $\Delta\Delta G$ prediction, *ISM* surpasses ESM2 in AUC (0.76 vs 0.72) and even outperforms specialized models that use atomic environments (0.76 vs 0.75, see Table 2). *ISM* structure-tunes ESM2 and can be quickly adopted by loading an *ISM* checkpoint into any pre-existing framework built on ESM2.

Sequence models trained using masked language modeling learn coarse structure features encoded in evolutionary co-variations, but these representations do not match the performance of a structure predictor that explicitly uses MSAs (Lin et al., 2022). This demonstrates that ESM2 representations fail to extract all structural information present within an MSA. Rather than further extracting structural information from evolutionary data, we enrich ESM2's structural representation by directly

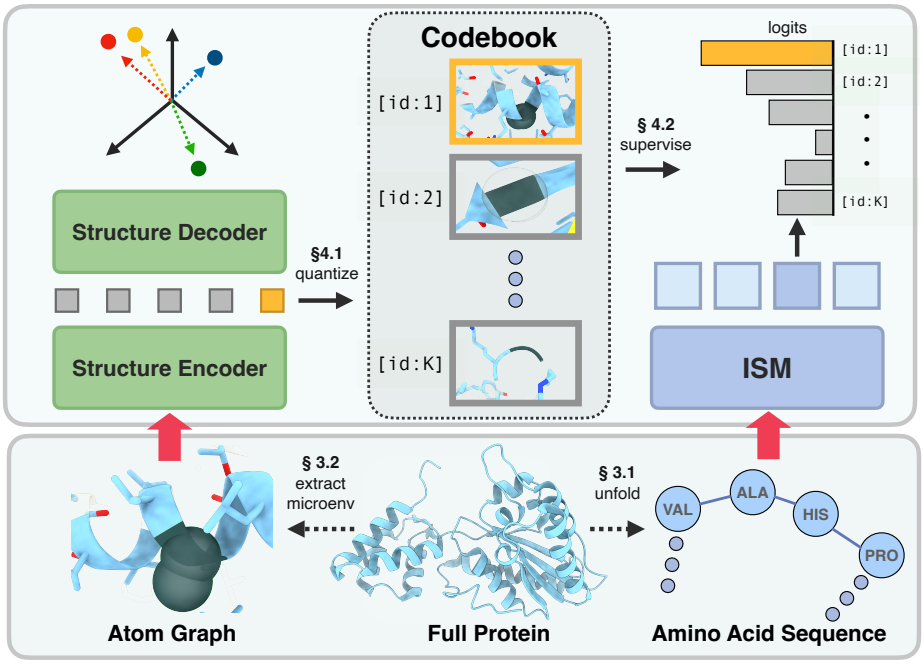


Figure 1: ***Implicit Structure Model* (*ISM*) is a sequence-only protein model (right) supervised by structure tokens derived from a structure model (left).** A structure encoder takes the atoms of a residue's microenvironment as input and produces a structural representation. We map this representation to a token in a codebook of structural motifs extracted via k-means clustering. The *ISM* sequence model learns to predict this structure token.

distilling representations from a structure model. *ISM* builds on a sequence-only model pre-trained on evolutionary data (i.e. ESM2) and is fine-tuned to predict a structure token – rather than a masked amino acid – for every residue (see Figure 1). A residue's structure token is derived from the representations of a structure model (i.e., Atomic Autoencoder in Section 4.1, MutRank in Gong et al. (2024)). This aligns with prior works which show that leveraging the interplay between multiple modalities, such as sequence and structure, enhances model performance (Gong et al., 2024; Hayes et al., 2024).

2 Related Work

Protein Language Models take an amino acid sequence as input and produce a deep representation for each amino acid conditioned on the entire sequence. Commonly-used models such as ProtBERT, ProteinBERT, ESM1b and ESM2 use transformer-based architectures and are trained to maximize wildtype accuracy (i.e., reconstruct masked amino acids) (Elnaggar et al., 2021; Brandes et al., 2022; Rives et al., 2019; Lin et al., 2022).

One of the motivations behind ESM2 was to build a single-sequence variant of AlphaFold that did not require the computationally expensive task of generating MSAs. The resulting model, ESMFold, is a widely used tool but generally underperforms when compared to AlphaFold. This demonstrates that the ESM2 does not fully capture the epistatic landscape imposed on evolution by the structure of a protein. This has led to various sequence models with a structural modality.

Sequence Models with structure loss The ESM2-s sequence model incorporates structural information by fine-tuning ESM2 to predict a protein's structural fold (Zhang et al., 2024). The fold, however, is coarse-grained information about the protein. *ISM* achieves superior performance by using the more fine-grained approach of training at the residue level. More specifically, in our training objective, each residue is tasked with predicting its corresponding local structural environment.

The "Structure-infused protein language models (SIPLM)" use a type of CLIP training to align sequence and structural features (Peñaherrera & Koes, 2023). This technique is also coarse-grained

because its training objective does not operate at a residue level (we do not include SIPLM in our tables of results due to its relatively weak performance on our benchmarks).

AlphaFold also learns structural representations from sequences (Jumper et al., 2021). However, it requires a multiple sequence alignment as input, which is expensive to compute and often unavailable for many practical applications. Furthermore, prior works have shown that Evoformer, the feature extractor for AlphaFold, underperforms ESM2 on various downstream tasks that involve less structural information (Hu et al., 2022). On these tasks, *ISM* still achieves comparable performance to ESM2.

Sequence models with structure inputs extend sequence models by making use of additional structural inputs. SaProt (Su et al., 2023) and ProstT5 (Heinzinger et al., 2023) use the VQ-VAE from FoldSeek (van Kempen et al., 2022) to extract per-residue structure tokens as additional inputs to the protein language model. MULAN (Frolova et al., 2024) extends these works to include structural features (torsion angles) as additional inputs to a protein language model (e.g., ESM2). Similarly, ProSST (Li et al., 2024) also takes structural tokens as inputs. Instead of using FoldSeek tokens, ProSST trains a Denoising Autoencoder to extract per-residue features, which are then tokenized into a structure sequence using K-means clustering. It then applies the amino acid and structure sequence of a protein as input to a transformer framework that makes use of a Sequence-Structure Disentangled Attention block. Nevertheless, ProSST also requires a protein structure as input at inference time.

ISM follows a similar structure tokenization scheme as ProSST but instead uses the structure tokens within an auxiliary loss to train additional classification heads. Thus, *ISM* does not require a structure as input at inference time.

Protein Structure Autoencoders take the backbone atom coordinates as input and encode each residue into a discrete token (Gaujac et al., 2024; Hayes et al., 2024). The sequence of discrete tokens then reconstructs atom positions, which are supervised using coordinate losses (e.g. frame aligned point error, histogram classification). Protein structure denoising Autoencoders take a noisy variant of the backbone as input and then learn a latent embedding that decodes the backbone (Peñaherrera & Koes, 2023; Li et al., 2024). Foldseek (van Kempen et al., 2022) extracts features for a residue given its nearest neighbor. These works use the protein backbone as input. In this work, we also train an Autoencoder but instead of reconstructing the local backbone of a protein, we reconstruct the coordinates of all atoms within the local chemical environment surrounding a masked residue (masked microenvironment).

3 Preliminaries

Let $\boldsymbol{x}_{\text{seq}} = (x_1, ..., x_L)$ be a protein sequence of L amino acids where each amino acid residue $x_l \in \{\text{A}, \text{C}, ..., \text{Y}\}$. The atoms defined by this sequence fold into an energetically favorable 3-dimensional structure $\boldsymbol{x}_{\text{struct}} = \{a_i = (p_i, e_i, \boldsymbol{c}_i)\}_{i=1}^{N}$ where each atom i consists of residue sequence position $p_i \in \{1, ..., L\}$, an element type $e_i \in \{\text{C}, \text{H}, \text{N}, \text{O}, \text{P}, \text{S}, \text{X}\}$ and coordinates $\boldsymbol{c}_i \in \mathbb{R}^3$. Let $\boldsymbol{\alpha}_l \in \mathbb{R}^3$ be the coordinate of the α-carbon atom for residue l.

3.1 Protein Sequence Models

A protein language models $\mathbf{PLM}$ takes a protein sequence $\boldsymbol{x}_{\text{seq}}$ as input and produces a latent representation $\mathbf{PLM}(\boldsymbol{x}_{\text{seq}}) \in \mathbb{R}^{L \times D}$ for downstream tasks. Most models use a transformer architecture and are pre-trained via a masked language modeling (MLM) loss. During training, a subset $\mathbb{M} \subset \{1, ..., L\}$ of the sequence is replaced with the `[mask]` token $\tilde{x}_i = \begin{cases} \texttt{[mask]} & \text{if } i \in \mathbb{M} \\ x_i & \text{otherwise} \end{cases}$ with $\tilde{\boldsymbol{x}}_{\text{seq}} = (\tilde{x}_1, ..., \tilde{x}_L)$. The model learns to reconstruct the masked tokens with

$$\mathcal{L}_{\text{MLM}} = \frac{1}{|\mathbb{M}|} \sum_{i \in \mathbb{M}} \ell_{\text{CE}}(\boldsymbol{C}_{\text{MLM}}^{\top} \mathbf{PLM}(\tilde{\boldsymbol{x}}_{\text{seq}})_i, x_i), \tag{1}$$

for the cross entropy loss ℓ_{CE}, indexed feature $\mathbf{PLM}(\boldsymbol{x}_{\text{seq}})_i \in \mathbb{R}^D$ at position i, and a linear classifier $\boldsymbol{C}_{\text{MLM}}$ that predicts the amino acid type. While the backbone $\mathbf{PLM}$ is used for downstream tasks, $\boldsymbol{C}_{\text{MLM}}$ is only used for pre-training.

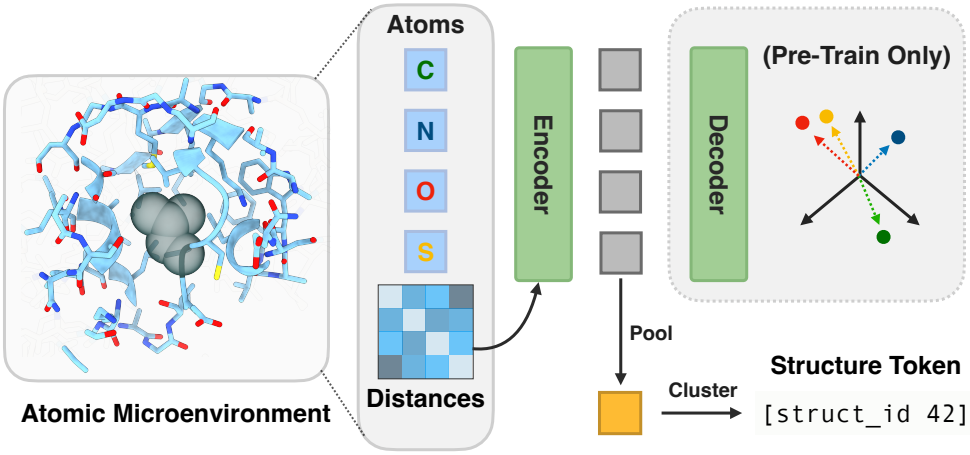


Figure 2: **Atomic Autoencoder tokenizes protein microenvironments to generate structure tokens that supervise sequence-only protein language model.** The Autoencoder takes atom element types and pairwise distances as input and reconstructs all atomic coordinates. The encoder is a graph transformer that uses the pairwise distances to bias the attention mechanism to learn rich atomic representations. The atomic representations are pooled to form a microenvironment embedding. The decoder takes the atomic representation and microenvironment embedding as input and produces coordinates for each atom. The microenvironment embedding is used to construct structural tokens for protein language model training. See Figure 3 for architectural details.

3.2 Protein Structure Models

A protein structure model **PSM** computes an atom-level feature representation from the local geometric description of each residue. It starts from a microenvironment $\boldsymbol{x}^l_{\text{microenv}}$ that contains all atoms in a radius $r = 8$Å around the α-carbon of residue l:

$$\boldsymbol{x}^l_{\text{microenv}} = \{(e_i, \boldsymbol{c}_i) : \forall i \in \{1, ..., N\} \text{ such that } ||\boldsymbol{c}_i - \boldsymbol{\alpha}_l|| < r\}.$$

A common backbone for protein structure models is a Graph Transformer G (Ying et al., 2021). The graph transformer $G(\boldsymbol{x}^l_{\text{microenv}})$ embeds each atom's element type e_i in a sequence $\mathbf{e} = \{e_1 \dots e_n\}$, where n is the size of the micro-environment. In attention updates, the graph transformer adds an attention bias $B^l_{ij} = \|\boldsymbol{c}_i - \boldsymbol{c}_j\|$ based on the pairwise distance between atoms i and j. This attention bias B^l is the only structural information given to the transformer. The graph transformer then produces a set of output features $\{\boldsymbol{z}^l_1 \dots \boldsymbol{z}^l_n\} = G(\boldsymbol{x}^l_{\text{microenv}})$, one per input atom e_i. The graph transformer is commonly trained on the end-task using a supervised learning objective (Ying et al., 2021). In this work, we use the Graph Transformer directly to train a structure model on atomic reconstructions of proteins in our pre-training dataset.

In MutComputeX-GT, Diaz et al. (2024) apply the Graph Transformer architecture to unsupervised pre-training using masked language modeling. They define a masked microenvironment $\boldsymbol{x}^l_{\text{masked-microenv}}$ that contains all atoms of other residues $p_i \neq l$

$$\boldsymbol{x}^l_{\text{masked-microenv}} = \{(e_i, \boldsymbol{c}_i) : \forall i \in \{1, ..., N\} \text{ such that } p_i \neq l \text{ and } ||\boldsymbol{c}_i - \boldsymbol{\alpha}_l|| < r\},$$

and pool all atom level features into a single residue level embedding $\boldsymbol{z}^l = \frac{1}{n}\sum_i \boldsymbol{z}^l_i$ for $\{\boldsymbol{z}^l_1 \dots \boldsymbol{z}^l_n\} = G(\boldsymbol{x}^l_{\text{masked-microenv}})$. They then use a masked language modeling objective to predict the masked out amino acid type:

$$\mathcal{L}^l_{\text{AA}} = \ell_{\text{CE}}(\boldsymbol{C}^\top_{\text{AA}} \boldsymbol{z}^l, x_l). \tag{2}$$

MutRank adds a self-supervised training objective to learn the evolutionary mutational landscape from the local structure (Gong et al., 2024) . More specifically, it learns an evolutionary score derived from the protein's multiple sequence alignment.

4 METHOD

ISM is a sequence model that takes as input only an amino acid sequence $\boldsymbol{x}_{\text{seq}}$ but is trained to implicitly capture structural information. We start by training an Atomic Autoencoder, based on Graph Transformer, on protein structures. The autoencoder is trained with a geometric reconstruction loss and the MutComputeX-GT objective. We then cluster the resultant features into K structure tokens $\{1, ..., K\}$. We use the sequence $\boldsymbol{s} = (s_1, ..., s_L)$ of structure tokens $s_l \in \{1, ..., K\}$ as an additional supervisory signal for the sequence-only ***I**mplicit **S**tructure **M**odel* (*ISM*).

4.1 ATOMIC AUTOENCODER

The Atomic Autoencoder uses an encoder-decoder architecture with a Graph Transformer encoder and a plain transformer decoder. The encoder takes the masked microenvironment $\boldsymbol{x}^l_{\text{microenv}}$ as input and produces atomic representations $\{\boldsymbol{z}^l_1 \dots \boldsymbol{z}^l_n\}$. The decoder takes atomic representations in and produces features $\{\boldsymbol{f}^l_1 \dots \boldsymbol{f}^l_n\}$ which linearly project to atomic coordinates $\{\boldsymbol{c}_i : \forall (e_i, \boldsymbol{c}_i) \in \boldsymbol{x}^l_{\text{microenv}}\}$ (See Figure 2). This might seem like a trivial task, after all the inputs $\boldsymbol{x}^l_{\text{microenv}}$ contain the regression targets. However, since the Graph Transformer only uses relative positions, and only in an attention bias B^l, the prediction tasks are quite difficult and require reasoning about the local structure of the micro-environment.

To obtain a residue-level feature representation, we average the atom-level features of the Graph Transformer $\boldsymbol{z}^l = \frac{1}{n}\sum_i \boldsymbol{z}^l_i$ following Diaz et al. (2024). To train this representation, we add $\boldsymbol{z}^l$ into all atomic representations prior to the decoder. Mathematically, the transformer decoder takes $\{\boldsymbol{z}^l_1 + \boldsymbol{z}^l \dots \boldsymbol{z}^l_n + \boldsymbol{z}^l\}$ as input. We also found that adding this $\boldsymbol{z}^l$ directly to the decoder architecture improves training stability. See Section A for full architecture.

Training objective. One major challenge in training microenvironment Autoencoders is that microenvironments lack robust protein backbone coordinate frames that underpin full protein models (Jumper et al., 2021; Hayes et al., 2024; Dauparas et al., 2022). We empirically observe that vanilla MSE loss $\mathcal{L}^l_{\text{MSE}} = \frac{1}{n}\sum_i \|\hat{\boldsymbol{c}}^l_i - \boldsymbol{c}^l_i\|$ does not take the coordinate frame into account and overestimates the loss. Instead, we optimize the minimal MSE loss under the optimal coordinate frame. More specifically, we compute the rotation and translation $(\boldsymbol{R}^l, \boldsymbol{T}^l)$ that minimize the MSE loss using Kabsch algorithm (Kabsch, 1976; Umeyama, 1991) and rotate the ground truth coordinates before applying the MSE loss. Formally,

$$\mathcal{L}^l_{\text{MSE-aligned}} = \frac{1}{n}\sum_i \|\hat{\boldsymbol{c}}^l_i - (\boldsymbol{R}^l \boldsymbol{c}^l_i + \boldsymbol{T}^l)\|.$$

During training, we observe that naive optimization of the MSE-aligned loss results in convergence to a local optimum where all predicted coordinates lie on a 2-dimensional plane. Following AlphaFold (Jumper et al., 2021), we addressed the issue using distogram loss. Here, we use ESM3's distogram head by first computing $\boldsymbol{f}^l_{ij} = \boldsymbol{W}_a \boldsymbol{f}^l_i - \boldsymbol{W}^l_b \boldsymbol{z}^l_j$, where $\boldsymbol{W}_a, \boldsymbol{W}_b$ are linear adapters. We apply a binned distance loss

$$\mathcal{L}^l_{\text{disto}} = \frac{1}{n^2}\sum_{i,j} \ell_{\text{CE}}(\boldsymbol{C}^T_{\text{disto}} \boldsymbol{z}^l_{ij}, d^{\text{bin},l}_{ij}).$$

where $\boldsymbol{C}_{\text{disto}}$ is a linear classifier that predicts the distance bin $d^{\text{bin},l}_{ij}$ between atoms i and j.

During the first stage of training, we train with the distogram and masked modeling losses, $\mathcal{L}^l_{dist} + \mathcal{L}^l_{\text{AA}}$. During the second stage, we include $\mathcal{L}^l_{\text{MSE-aligned}}$.

Generating Structure Tokens. Given a protein structure $\boldsymbol{x}_{\text{struct}}$, we start by generating the masked microenvironment for all residues, namely $\{\boldsymbol{x}^1_{\text{microenv}} \dots \boldsymbol{x}^L_{\text{microenv}}\}$ where L is the number of amino acids in the protein. We feed each masked microenvironment into our Graph Transformer encoder to extract a residue-level feature representation at each position, $\{\boldsymbol{z}^1 \dots \boldsymbol{z}^L\}$. We quantize $\boldsymbol{z}^l$ for every residue in the protein using K-means (Lloyd, 1982) to generate a structure sequence $\boldsymbol{s} = (s_1, ..., s_L)$. In addition to our autoencoder, we also extract features $\{\boldsymbol{z}^{1\prime} \dots \boldsymbol{z}^{L\prime}\}$ from EvoRank (Gong et al., 2024) and generate a second structure sequence $\boldsymbol{s}' = (s'_1, ..., s'_L)$, both of which are used to fine-tune the protein sequence model. Both models are trained on a smaller dataset of experimental structures and run on a large dataset of AlphaFold structures.

4.2 STRUCTURE-TUNING THE PROTEIN SEQUENCE MODEL

We initialize a sequence-only protein language model trained using masked language modeling, *i.e.* ESM2, and continue fine-tuning it to predict the structure tokens. We call this training **structure-tuning** and the resulting final model **Implicit Structure Model**. In contrast to ProtBert and ESM2, which predict residues at masked positions, we find that predicting structure tokens at *all* positions better distills structural representations. We append a linear classifier $\boldsymbol{C}_{\text{struct}}$ to the output of the pLM backbone to predict the structural token. The structure prediction loss function is

$$\mathcal{L}_{\text{Struct}} = \frac{1}{L}\sum_{i=1}^{L} \ell_{\text{CE}}(\boldsymbol{C}_{\text{struct}}^{\top}\mathbf{PLM}(\tilde{\boldsymbol{x}}_{\text{seq}})_i, s_i),$$

where $\tilde{\boldsymbol{x}}_{\text{seq}}$ is the amino acid sequence with masked residues, $\mathbf{PLM}$ is the pLM backbone, $\mathbf{PLM}(\tilde{\boldsymbol{x}}_{\text{seq}})_i$ is the representation for residue i, and s_i is the structure token at residue i.

We perform structure-tuning on 6M protein sequences and their corresponding AlphaFold or experimental structure (Ahdritz et al., 2022). Our structure tokens not only capture rich structural motifs but also help us identify and discard microenvironments of poor structural quality. More specifically, we find that residues in poorly folded regions within an AlphaFold structure (*i.e.* low pLDDT score, lack of secondary and tertiary structure) tend to collapse to a specific structure token s^* (visualized as `[struct id 17]` in Figure 5). Thus, we do not supervise the sequence model with microenvironments assigned the s^* token. Our revised structure-tuning loss is

$$\mathcal{L}_{\text{Struct}} = \frac{1}{|\mathbb{S}|}\sum_{i\in\mathbb{S}} \ell_{\text{CE}}(C_{\text{struct}}^{\top}\mathbf{PLM}(\tilde{\boldsymbol{x}}_{\text{seq}})_i, s_i),$$

where $\mathbb{S} = \{i : i \in [1, L] \text{ and } s_i \neq s^*\}$.

The final training objective for structure-tuning is the sum of structure token and amino acid cross-entropy losses (see Section 3.1), namely $\mathcal{L} = \mathcal{L}_{\text{Struct}} + \mathcal{L}_{\text{MLM}}$.

5 RESULTS

5.1 IMPLEMENTATION DETAILS

Microenvironment autoencoder. Our autoencoder is a Graph Transformer encoder with 4 layers and a vanilla Transformer decoder with 2 layers. Our autoencoder training dataset contains 30,000 proteins from the Protein Data Bank(PDB). We train both stages for 5 epochs with a learning rate of 1e-3. See Table 8a for a list of hyperparameters.

Distillation Dataset. Once our model is fully trained, we extract microenvironment features for 7M proteins from Uniclust30 with AlphaFold structures (Mirdita et al., 2017), along with the training set of 30K PDB proteins. We apply K-means only to the PDB subset. The number of clusters, $K = 64$, is chosen using the elbow method. Additionally, we extract microenvironment features from MutRank and extract $K = 512$ centroids from the PDB proteins (see Section 3.2).

Structure-tuning. We structure-tune the 650 M parameter ESM2 for 20 epochs using a cosine learning rate schedule with 4 warmup epochs. We use a batch size of 48 proteins cropped to a maximum sequence length of 512 amino acids. We use AdamW optimizer with a learning rate of 1×10^{-4} and weight decay of 5×10^{-3}. Training takes 26 wall-clock hours on 32 GH200 GPUs. See Table 8b for a complete list of hyperparameters.

5.2 COMPARISONS ON STRUCTURE TASKS

In Table 1, we evaluate the structure-enriched representation of *ISM* against established methods on several structure-based downstream tasks, including structure, contact, secondary structure, and binding residue prediction. For structure prediction, we initialize from pre-trained SoloSeq (Ahdritz et al., 2022), replace the backbone with a frozen *ISM* and tune the folding head. For other downstream tasks, we freeze the backbone and train a linear head. Dataset descriptions are listed in Section D. We report the performance of a fine-tuned ESM which follows the same training regimen as *ISM*, but uses only cross entropy loss on the masked amino acid. We report results for models trained on Uniclust30 alone and Uniclust30+PDB.

Table 1: **Comparisons on structural benchmarks.** We freeze all model backbones to assess the learned representation. *ISM* is structure-tuned on structural tokens obtained from the AlphaFold structures of Uniclust30 while ISM† undergoes additional structure-tuning with structure tokens obtained from PDB structures. SaProt* takes the protein structure as input. The *ISM* framework generates structure-enriched representations using the architecture of ESM2.

	Structure Prediction (CAMEO)			Contact			SS	Binding	
Method	GDT-TS	GDT-HA	LDDT	Short	Med	Long	Acc	F1	MCC
Evolutionary pLM									
ESM2 (Lin et al., 2022)	0.64	0.47	0.82	0.45	0.45	0.35	0.86	0.31	0.34
ESM2 (fine-tuned)	0.64	0.47	0.82	0.45	0.45	0.35	0.86	0.32	0.34
Structural pLM									
ESM2-S (Zhang et al., 2024)	0.61	0.43	0.79	0.46	0.47	0.36	0.85	0.32	0.35
SaProt* (Su et al., 2023)	-	-	-	0.57	0.53	0.48	0.86	0.36	**0.38**
ISM (Ours)	**0.67**	**0.50**	0.83	0.61	**0.60**	**0.49**	**0.89**	0.35	0.37
ISM † (Ours)	**0.67**	**0.50**	**0.84**	**0.62**	**0.60**	0.48	**0.89**	**0.37**	**0.38**

Table 2: **System-level Comparisons on S669 Single Mutation Thermodynamic Stability prediction.** We report regression and classification metrics. Our fine-tuning training set, cDNA117K, consists of mini-proteins that have at most 30% sequence similarity with those in S669. Top block shows comparisons with literature methods. Middle block shows structure and sequence-based approaches also fine-tuned on cDNA117k. UR50: UniRef-50 used in ESM2 pretraining, PDB: Protein data bank, UC30: Uniclust30. r_s: Spearman correlation coefficient.

Method	PreTrain Data	r_s	AUC	MCC	RMSE$_{\downarrow}$
FoldX (Schymkowitz et al., 2005)	N/A	0.27	0.62	0.14	2.35
PROSTATA (Umerenkov et al., 2022)	UR-50	0.50	0.73	0.28	1.44
Stability Oracle (Diaz et al., 2024)	PDB	0.53	0.75	0.34	1.44
MutateEverything (ESM) (Ouyang-Zhang et al., 2024)	UR-50	0.47	0.72	0.31	1.48
MutateEverything (AF) (Ouyang-Zhang et al., 2024)	PDB	**0.56**	**0.76**	0.35	**1.38**
SaProt (Su et al., 2023)	UR50,UC30	0.49	0.71	0.25	1.47
ESM3 (Hayes et al., 2024)	UR70,PDB,MGnify,JGI,OAS,AFDB,ESMAtlas	0.46	0.70	0.26	1.49
ESM (ft)	UR-50,PDB+UC30	0.49	0.72	0.25	1.47
ISM (MutRank only)	UR50,PDB+UC30	0.51	0.74	0.33	1.45
ISM (MutRank$\times$2)	UR50,PDB+UC30	0.50	0.73	0.32	1.45
ISM	UR50,UC30	0.49	0.73	0.33	1.47
ISM	UR50,PDB	0.52	0.74	0.30	1.45
ISM (Ours)	UR50,PDB+UC30	0.53	**0.76**	**0.40**	1.44

Our model outperforms all sequence-only and structural sequence models on all structure-based benchmarks. Notably, on structure prediction *ISM* outperforms ESM2 by 5% on the GDT-TS metric: 0.67 vs 0.64. On binding residue prediction F1 metric, *ISM* performs similarly with SaProt's 0.36, achieving 0.35 when trained on Uniclust30 and 0.37 when trained on Uniclust30+PDB. We note that SaProt explicitly requires the structure as input to achieve 0.36 while *ISM* is sequence only. Overall, the structure-enriched representations of *ISM* improve performance on various structure-based downstream tasks compared to sequence-only pLMs and structural pLMs.

5.3 Comparisons on Mutation Stability Effect

We evaluate how effectively *ISM* predicts the impact of mutations on a protein's thermodynamic stability ($\Delta\Delta G$). Table 2 shows *ISM* performance against existing work on the S669 single mutations dataset (Pancotti et al., 2022). We fine-tune on the cDNA117K dataset from Diaz et al. (2024), a subset of the cDNA display proteolysis dataset (Tsuboyama et al., 2023) where all proteins have at most 30% sequence similarity to those in S669. *ISM* outperforms all existing models that take a single sequence as input, achieving a Spearman correlation of 0.53 compared to ESM's 0.49, and an AUC of 0.76 compared to ESM's 0.72. Additionally, *ISM* matches the performance of state-of-the-art models while only using the amino acid sequence input, achieving an AUC of 0.76, while Mutate Everything and Stability Oracle achieve AUCs of 0.76 and 0.75, respectively. ISM also runs 20$\times$ faster on a protein of 300 amino acids. Note that Stability Oracle (Diaz et al., 2024) takes the atomic microenvironment as input and Mutate Everything-AF (Ouyang-Zhang et al., 2024) takes a multiple sequence alignment as input.

Table 3: **System-level Comparisons to prior work on various functional benchmarks.** Our method is fine-tuned for all benchmarks except HumanPPI, in which we freeze the backbone and perform linear probing. * indicates the best checkpoint taken during training.

Method	Thermostability	HumanPPI	Metal Bind	EC	GO			DeepLoc	
					MF	BP	CC	Subcell.	Binary
	Spearman ρ	Acc	Acc	Fmax	Fmax	Fmax	Fmax	Acc	Acc
ESM1b	0.71	0.82	0.74	0.87	0.66	0.45	0.47	0.80	0.92
MIF-ST	0.69	0.76	0.75	0.81	0.63	0.38	0.32	0.79	0.92
ESM2*	0.70	0.88	0.74	0.87	**0.67**	**0.49**	0.51	**0.85**	**0.94**
SaProt*	**0.72**	0.88	**0.79**	**0.88**	0.65	**0.49**	0.51	**0.85**	0.93
ISM *	0.71	**0.89**	0.75	**0.88**	**0.67**	0.47	**0.52**	0.84	0.93

We conducted an ablation study on the datasets used for structure-tuning and were surprised to find that training on the smaller PDB dataset enhances downstream $\Delta\Delta G$ performance more than training on the larger Uniclust30 dataset. Specifically, *ISM* achieves a Spearman correlation of 0.49 when trained on UniClust30, compared to 0.52 when trained on PDB. Even though the supervision signal during structure-tuning is derived solely from the atomic coordinates in the structure and not $\Delta\Delta G$ labels, we suspect the PDB dataset has some overlap with the structures in the S669 dataset, resulting in performance similar to that of structure-input models. Overall, on the S669 $\Delta\Delta G$ test set, *ISM* is competitive and even outperforms SOTA structure-based methods and AlphaFold's representations, a feat sequence-only pLMs have yet to achieve.

5.4 Comparisons on a diverse set of functional phenotypes

In Table 3, we evaluate *ISM* on the PEER (Xu et al., 2022) and FLIP (Dallago et al., 2021) benchmarks, which encompass tasks that benefit from structural representations (e.g., thermostability), evolutionary representations (e.g., biological process), or both (e.g., EC). We fine-tune ESM2 and *ISM* on all benchmarks, except HumanPPI, for which we perform linear probing to prevent overfitting. We observed that longer training leads to overfitting, therefore, we evaluate various training checkpoints and report the highest performance for ESM2, SaProt, and *ISM*. Metrics of ESM1b (Rives et al., 2019) and MIF-ST (Yang et al., 2023) are sourced from SaProt (Su et al., 2023).

We observe that while *ISM* performance remains competitive with ESM2 and other pLMs on functionally diverse tasks and does not stand out. For example, for predicting gene ontology - molecular function, both *ISM* and ESM2 achieve 67% accuracy while SaProt achieves 65%. This finding aligns with prior work (Hu et al., 2022), which demonstrates that ESM2 outperforms Evoformer, the feature extractor for AlphaFold, on some functional tasks. It seems that for these functional tasks, the evolutionary signal from masked amino acid modeling is sufficient and does not necessarily benefit from structurally-enriched representations. Nonetheless, these experiments demonstrate that the structure-enriched representations of *ISM* do not corrupt ESM2's evolutionary representation on various function-based downstream tasks while enhancing their understanding of structure.

6 Analysis

6.1 Ablations

We ablate key design decisions and report long-range Precision at L for contact prediction, accuracy for secondary structure prediction, F1 for binding residue prediction, and Spearman correlation for mutation stability effect prediction in Table 4. We also report the validation accuracy, indicating how often the *ISM* variant correctly predicts the structure token derived from the atomic autoencoder.

Structure Tokens. In Table 4a, we distill from various structure models from the literature. We compare against a variant using MutRank and MutCompute structure models. Since Atomic Autoencoder uses the loss $\mathcal{L}^{l}_{\text{AA}}$ from MutCompute, this variant effectively removes the autoencoder from structure-tuning. Our model outperforms MutRank and MutCompute, indicating that the autoencoder provides important structural information.

Table 4: ***ISM* ablation experiments.** Default Settings are marked in grey. See Section 6.1. ss: Secondary Structure prediction, mc: MutCompute, mr: MutRank, ae: Autoencoder

(a) **Other Structure Tokens**

tokenizer	contact	ss	bind
foldseek	0.42	0.88	0.32
esm3	0.18	0.85	0.11
mc+mr	0.45	0.88	0.36
ae+mr	**0.48**	**0.89**	**0.37**

(b) **Our Structure Tokens**

tokenizer	contact	ss	bind
ae	0.38	0.88	0.35
mr	0.46	0.88	0.34
mr $\times$ 2	**0.52**	0.88	**0.36**
ae+mr	0.49	**0.89**	0.35

(c) **Number of clusters**

K	contact	ss	bind
32	0.27	0.84	0.33
64	**0.48**	**0.89**	**0.37**
128	0.42	0.85	**0.37**

(d) **Pre-training Crop length**

crop	val acc	contact
32	0.27	0.27
128	0.36	0.42
512	**0.40**	**0.48**

(e) **Label Type**

label	contact	r_s ($\Delta\Delta G$)
features	0.36	0.49
tokens	**0.46**	**0.51**

(f) **Initialization**

init	val acc	contact
random	0.36	0.10
esm2	**0.40**	**0.48**

We found that using ESM3's VQVAE (Hayes et al., 2024) structure tokens for structure-tuning does not produce robust structural representations. In long-range P@L contact prediction and the binding residue tasks, the F1 metrics are 0.18 and 0.11 compared to 0.48 and 0.37 for *ISM*, respectively. We observe that the accuracy of ESM3 VQ-VAE structure token prediction on a held-out validation accuracy on UniClust30 is around 8% (Autoencoder accuracy is $\sim$40% and MutRank accuracy is $\sim$47%). We suspect that the large vocabulary of ESM3 VQ-VAE (4096 structure tokens) results in redundant and overlapping tokens that are difficult to discern and complicate loss optimization.

We also evaluate the performance of our sequence model using FoldSeek VQ-VAE structure tokens for structure-tuning (van Kempen et al., 2022). We train on a larger subset of UniClust30 obtained from SaProt (Su et al., 2023), using the same number of iterations as *ISM*. The model achieves a long-range contact P@L of 0.42 and a binding residue F1 score of 0.32, which are improvements over ESM3 VQ-VAE structure tokens and surpasses the ESM2 baseline (F1 scores of 0.35 and 0.31, respectively). However, representations learned from FoldSeek's VQ-VAE structure tokens lag behind *ISM* (0.48 and 0.37). Thus, the structure tokens from our Autoencoder and MutRank produce better structure representations, their combination being the most effective (see Table 4b).

Training parameters. We evaluate how much the maximum length of a sequence during structure-tuning affects the structure accuracy and downstream performance in Table 4d. We find that when the crop length is dropped to 128 and 32 amino acids, the contact long-range P@L drops from 0.48 to 0.42 and 0.27 respectively. This shows that training with longer sequences is essential for learning long-range contacts. Additionally, we evaluate the effectiveness of clustering MutRank representations into tokens in Table 4e (excluding the Autoencoder supervision). Our model variant uses a linear head to predict features and is trained using cosine distance to MutRank representations. Direct feature prediction achieves 0.36 P@L, while cluster ID prediction reaches 0.46 P@L on long-range contact prediction. Clustering potentially removes superfluous high-frequency noise.

Evolutionary Pre-Training. We evaluate the significance of MLM as a pre-training stage before structure tuning in Table 4f by initializing with random weights. This approach resulted in decreased accuracy of structure tokens from 40% to 36%. On downstream contact prediction, training from scratch drops long-range P@L from 0.48 to 0.1. This highlights the value of structure-tuning ESM2 evolutionary representations over training from scratch.

7 Conclusions

We introduce a novel pre-training stage for protein language models to learn enhanced structural representations. We supervise the protein language model with structure tokens extracted from structure models. *ISM*'s structural representations improve performance across a variety of structural benchmarks including structure, contact, secondary structure, and binding prediction. *ISM* is a one-line code replacement in any framework built using ESM2.

References

Gustaf Ahdritz, Nazim Bouatta, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, Bo Zhang, Arkadiusz Nowaczynski, Bei Wang, Marta M Stepniewska-Dziubinska, Shang Zhang, Adegoke Ojewole, Murat Efe Guney, Stella Biderman, Andrew M Watkins, Stephen Ra, Pablo Ribalta Lorenzo, Lucas Nivon, Brian Weitzner,

Yih-En Andrew Ban, Peter K Sorger, Emad Mostaque, Zhao Zhang, Richard Bonneau, and Mohammed AlQuraishi. Openfold: Retraining alphafold2 yields new insights into its learning mechanisms and capacity for generalization. *bioRxiv*, 2022. doi: 10.1101/2022.11.20.517210. 6

Christian B Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096): 223–230, 1973. 1

Nadav Brandes, Dan Ofer, Yam Peleg, Nadav Rappoport, and Michal Linial. Proteinbert: a universal deep-learning model of protein sequence and function. *Bioinformatics*, 38(8):2102–2110, 2022. 1, 2

Christian Dallago, Jody Mou, Kadina E Johnston, Bruce J Wittmann, Nicholas Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. *bioRxiv*, pp. 2021–11, 2021. 8

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022. 5

D. J. Diaz, C. Gong, J. Ouyang-Zhang, J. M. Loy, J. Wells, D. Yang, A. D. Ellington, A. G. Dimakis, and A. R. Klivans. Stability oracle: A structure-based graph-transformer for identifying stabilizing mutations. *Nature Communications*, 15:6170, 2024. doi: 10.1038/s41467-024-49780-2. 4, 5, 7

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021. 1, 2

Daria Frolova, Marina Pak, Anna Litvin, Ilya Sharov, Dmitry Ivankov, and Ivan Oseledets. Mulan: Multimodal protein language model for sequence and structure encoding. *bioRxiv*, pp. 2024–05, 2024. 1, 3

Benoit Gaujac, Jérémie Donà, Liviu Copoiu, Timothy Atkinson, Thomas Pierrot, and Thomas D Barrett. Learning the language of protein structure. *arXiv preprint arXiv:2405.15840*, 2024. 1, 3

Chengyue Gong, Adam Klivans, James Madigan Loy, Tianlong Chen, Daniel Jesus Diaz, et al. Evolution-inspired loss functions for protein representation learning. In *Forty-first International Conference on Machine Learning*, 2024. 2, 4, 5

Tomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *bioRxiv*, pp. 2024–07, 2024. 2, 3, 5, 7, 9

Michael Heinzinger, Konstantin Weissenow, Joaquin Gomez Sanchez, Adrian Henkel, Milot Mirdita, Martin Steinegger, and Burkhard Rost. Bilingual language model for protein sequence and structure. *bioRxiv*, pp. 2023–07, 2023. 3

Mingyang Hu, Fajie Yuan, Kevin Yang, Fusong Ju, Jin Su, Hui Wang, Fei Yang, and Qiuyang Ding. Exploring evolution-aware &-free protein language models as protein function predictors. *Advances in Neural Information Processing Systems*, 35:38873–38884, 2022. 3, 8

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. 3, 5

Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976. 5

Mingchen Li, Yang Tan, Xinzhu Ma, Bozitao Zhong, Huiqun Yu, Ziyi Zhou, Wanli Ouyang, Bingxin Zhou, Liang Hong, and Pan Tan. Prosst: Protein language modeling with quantized structure and disentangled attention. *bioRxiv*, pp. 2024–04, 2024. 1, 3

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022. 1, 2, 7

Maria Littmann, Michael Heinzinger, Christian Dallago, Konstantin Weissenow, and Burkhard Rost. Protein embeddings and deep learning predict binding residues for various ligand classes. *Scientific Reports*, 11(1):23916, 2021. 14

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982. 5

Milot Mirdita, Lars Von Den Driesch, Clovis Galiez, Maria J Martin, Johannes Söding, and Martin Steinegger. Uniclust databases of clustered and deeply annotated protein sequences and alignments. *Nucleic acids research*, 45(D1):D170–D176, 2017. 6

Milot Mirdita, Konstantin Schütze, Yoshitaka Moriwaki, Lim Heo, Sergey Ovchinnikov, and Martin Steinegger. Colabfold: making protein folding accessible to all. *Nature methods*, 19(6):679–682, 2022. 13

Jeffrey Ouyang-Zhang, Daniel Diaz, Adam Klivans, and Philipp Krähenbühl. Predicting a protein's stability under a million mutations. *Advances in Neural Information Processing Systems*, 36, 2024. 7

Corrado Pancotti, Silvia Benevenuta, Giovanni Birolo, Virginia Alberini, Valeria Repetto, Tiziana Sanavia, Emidio Capriotti, and Piero Fariselli. Predicting protein stability changes upon single-point mutation: a thorough comparison of the available tools on a new dataset. *Briefings in Bioinformatics*, 23(2):bbab555, 2022. 7

Daniel Peñaherrera and David Ryan Koes. Structure-infused protein language models. *bioRxiv*, 2023. 2, 3

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019. 14

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *PNAS*, 2019. doi: 10.1101/622803. 1, 2, 8

Joost Schymkowitz, Jesper Borg, Francois Stricher, Robby Nys, Frederic Rousseau, and Luis Serrano. The foldx web server: an online force field. *Nucleic acids research*, 33(suppl_2):W382–W388, 2005. 7

Jin Su, Chenchen Han, Yuyang Zhou, Junjie Shan, Xibin Zhou, and Fajie Yuan. Saprot: Protein language modeling with structure-aware vocabulary. *bioRxiv*, 2023. 1, 3, 7, 8, 9, 13, 14

Kotaro Tsuboyama, Justas Dauparas, Jonathan Chen, Elodie Laine, Yasser Mohseni Behbahani, Jonathan J Weinstein, Niall M Mangan, Sergey Ovchinnikov, and Gabriel J Rocklin. Mega-scale experimental analysis of protein folding stability in biology and design. *Nature*, 2023. doi: 10.1038/s41586-023-06328-6. URL https://doi.org/10.1038/s41586-023-06328-6. 7

Dmitriy Umerenkov, Tatiana I Shashkova, Pavel V Strashnov, Fedor Nikolaev, Maria Sindeeva, Nikita V Ivanisenko, and Olga L Kardymon. Prostata: Protein stability assessment using transformers. *bioRxiv*, pp. 2022–12, 2022. 7

Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04):376–380, 1991. 5

Michel van Kempen, Stephanie S Kim, Charlotte Tumescheit, Milot Mirdita, Cameron LM Gilchrist, Johannes Söding, and Martin Steinegger. Foldseek: fast and accurate protein structure search. *Biorxiv*, pp. 2022–02, 2022. 3, 9, 13

Minghao Xu, Zuobai Zhang, Jiarui Lu, Zhaocheng Zhu, Yangtian Zhang, Ma Chang, Runcheng Liu, and Jian Tang. Peer: a comprehensive and multi-task benchmark for protein sequence understanding. *Advances in Neural Information Processing Systems*, 35:35156–35173, 2022. 8, 14

Jianyi Yang, Ambrish Roy, and Yang Zhang. Biolip: a semi-manually curated database for biologically relevant ligand–protein interactions. *Nucleic acids research*, 41(D1):D1096–D1103, 2012. 14

Kevin K Yang, Niccolò Zanichelli, and Hugh Yeh. Masked inverse folding with sequence transfer for protein representation learning. *Protein Engineering, Design and Selection*, 36:gzad015, 2023. 1, 8

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021. 4

Zuobai Zhang, Jiarui Lu, Vijil Chenthamarakshan, Aurélie Lozano, Payel Das, and Jian Tang. Structure-informed protein language model. *arXiv preprint arXiv:2402.05856*, 2024. 1, 2, 7

A ATOMIC AUTOENCODER ARCHITECTURE DETAILS

In Figure 3, we visualize the details of our atomic autoencoder architecture. We use a GraphTransformer encoder and a vanilla transformer decoder.

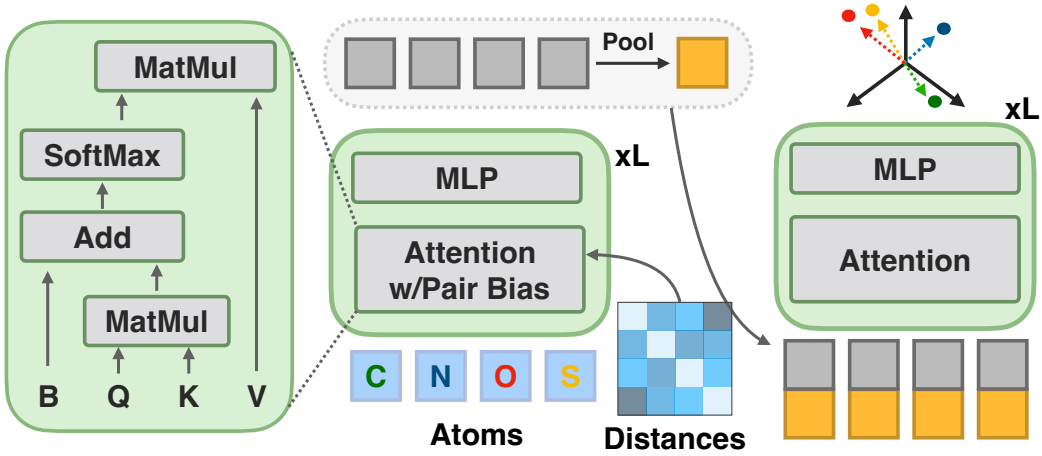


Figure 3: **Atomic Autoencoder Architecture Details.** The Autoencoder takes atom element types and pairwise distances as input and reconstructs all atomic coordinates. The encoder is a graph transformer that uses the pairwise distances to bias the attention mechanism to learn rich atomic representations. The atomic representations are pooled to form a microenvironment embedding. The decoder takes the atomic representation and microenvironment embedding as input and produces coordinates for each atom. The microenvironment embedding is used to construct structural tokens for protein language model training.

B ATOMIC AUTOENCODER DATASET

We downloaded a list of proteins from the PDB via PISCES (https://dunbrack.fccc.edu/pisces/) on October 23rd, 2023. We use the 95% sequence similarity split with 37,907 protein chains. We keep all proteins resolved by X-ray crystallography with resolution better than 3Å with no residue breaks and sequence length between 40 and 10,000. After our data pipeline and additional filtering, we ended up with 35,985 proteins in our PDB training set.

C RUNTIME

We compare our runtime against SaProt (Su et al., 2023) on three proteins with 91, 355, and 689 amino acids. We use ColabFold (Mirdita et al., 2022) to obtain an AlphaFold structure and use FoldSeek (van Kempen et al., 2022) to tokenize the structure. The transformer forward pass is performed on an A40 GPU. On average, Colabfold took 418 seconds while Foldseek and transformer forward pass took 43 and 28 milliseconds. By far, Colabfold structure prediction dominates the runtime. Even with structures, the SaProt runtime is about 2.4× slower than the *ISM* pipeline.

D Structural Benchmark Details

D.1 Structure Prediction

We train on proteins in the PDB and evaluate on the CAMEO dataset. We freeze *ISM* and train a folding trunk for 10 epochs using a cosine learning rate schedule with 2 warmup epochs. We use a batch size of 128 proteins. We use LION optimizer with a learning rate of 1×10^{-4} and weight decay of 0.01.

We additionally include comparisons to SoloSeq and AlphaFold below.

Table 5: **System-level Comparisons to prior work on CAMEO structure prediction.**

Method	GDT-TS	GDT-HA	LDDT
SoloSeq	0.61	0.43	0.79
with ESM2	0.64	0.47	0.82
with *ISM*	0.67	0.50	0.83
AlphaFold2	0.75	-	0.89

D.2 Contact Prediction

We follow the experimental setting as in SaProt (Su et al., 2023), which uses the contact prediction benchmark proposed by Rao et al. (2019); Xu et al. (2022). In the main paper, we report precision at L (P@L) for long-range contacts at least 24 amino acids away. In Table 6, we thoroughly evaluate precision at L, L/2, L/5 on short, medium, and long-range intervals of [6,12], [12,24],[24,∞] amino acids respectively.

Table 6: **System-level Comparisons to prior work on contact prediction.**

	Short Range			Medium Range			Long Range		
Method	P@L	P@L/2	P@L/5	P@L	P@L/2	P@L/5	P@L	P@L/2	P@L/5
ESM-2	0.45	0.45	0.50	0.45	0.45	0.54	0.35	0.42	0.52
ESM-2S	0.46	0.46	0.50	0.46	0.47	0.54	0.36	0.43	0.52
SaProt†	0.57	0.57	0.64	0.53	0.55	0.66	0.48	**0.60**	**0.74**
ISM (Ours)	**0.62**	**0.62**	0.67	**0.60**	**0.61**	**0.68**	**0.49**	0.57	0.69
ISM † (Ours)	**0.62**	**0.62**	**0.68**	**0.60**	0.60	**0.68**	0.48	0.56	0.67

D.3 Secondary Structure

We use the secondary structure prediction benchmark from Xu et al. (2022) in which secondary structures are labeled as either coil, strand, or helix. The maximum sequence similarity between a protein in the training and test set is 25%. We evaluate the model's accuracy.

We freeze *ISM* and train a linear classifier for 10 epochs using a cosine learning rate schedule with 2 warmup epochs. We use a batch size of 32 proteins. We use AdamW optimizer with a learning rate of 1×10^{-4} and weight decay of 0.5.

D.4 Binding Residues

We use the binding residues benchmark extracted from BioLip (Yang et al., 2012) prepared in the bindEmbed21 method (Littmann et al., 2021). It involves binary classification of whether a residue is within < 2.5Å of a metal ion, nucleic acid, and/or a ligand (Littmann et al., 2021). We freeze *ISM* and train a linear classifier for 10 epochs using a cosine learning rate schedule with 2 warmup epochs. We use a batch size of 32 proteins. We use AdamW optimizer with a learning rate of 3×10^{-4} and weight decay of 0.5. Full results are available in Table 7.

Table 7: **System-level Comparisons to prior work on binding residue prediction.**

Method	Test F1	Test MCC	Test AUC	Independent F1	Independent MCC	Independent AUC
ESM	0.31	0.34	0.84	0.28	0.28	0.82
ESM-2S	0.32	0.35	0.84	0.28	0.28	0.83
SaProt †	0.36	**0.38**	**0.87**	**0.35**	**0.33**	**0.87**
ISM (Ours)	0.35	0.37	0.86	0.33	0.31	0.85
ISM † (Ours)	**0.37**	**0.38**	0.86	0.34	0.32	0.85

E TRAINING DETAILS

Table 8 lists the hyperparameters for training Automatic Autoencoder (Section 4.1) and structure-tuning the PLM (Section 4.2). In Table 9, we present the hyperparameters for fine-tuning on different downstream benchmarks in Section D.

Table 8: **Hyperparameters for training.** Here we show the hyperparameters used to train the autoencoder (left) and *ISM* (right).

(a) **Atomic Autoencoder Training**

Hyperparameter	**Stage 1**	**Stage 2**
optimization		
total batch size	2048	2048
optimizer	AdamW	AdamW
learning rate	1e-3	1e-3
weight decay	1e-5	1e-5
epochs	5	5
warmup epochs	1	1
clip max norm	1.0	1.0
layers	4	4
number of GPUs	8	8
max atoms	512	512
max atom dist	8.0	8.0
λ_{AA}	1.0	1.0
$\lambda_{\text{Distogram}}$	1.0	1.0
$\lambda_{\text{MSE-aligned}}$	0	1.0
$\lambda_{\text{Distances}}$	0	1.0

(b) **Protein Language Model Structure Tuning**

Hyperparameter	**Structure-tuning**
optimization	
total batch size	1536
optimizer	AdamW
learning rate	1e-4
weight decay	5e-3
epochs	20
warmup epochs	4
clip max norm	5.0
layers	33
number of GPUs	32
mask ratio	15%
crop length	512
λ_{MLM}	1.0
λ_{struct1}	1.0
λ_{struct2}	1.0

Table 9: **Hyperparameters for structure-tuning on different benchmarks.** ⋆: we find that training converges and terminates training early.

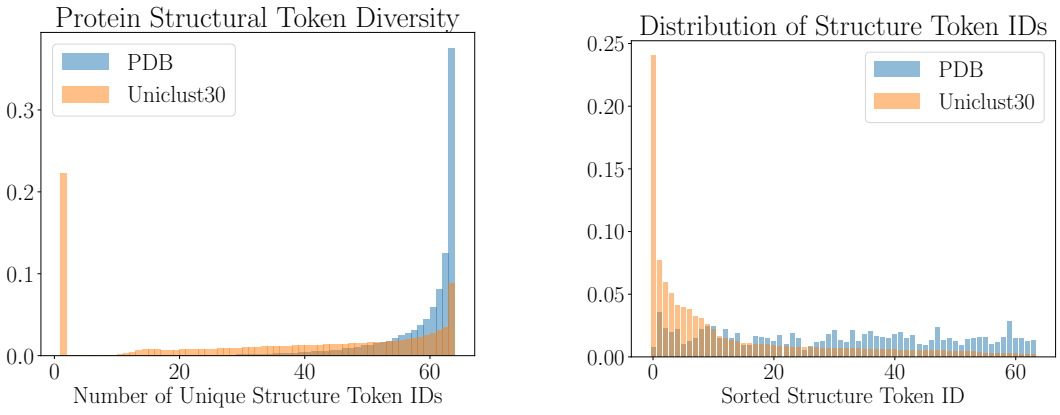
Hyperparameter	**Structure**	**Contact**	**Secondary Structure**	**Binding Residues**
total batch size	128	16	16	32
optimizer	LION	AdamW	AdamW	AdamW
learning rate	1e-4	0.01	3e-4	1e-4
weight decay	5e-3	0.01	0.5	0.5
epochs	20	30	10	10
warmup epochs	4	-	2	2
clip max norm	5.0	-	5.0	5.0
freeze backbone	True	True	True	True
number of GPU	32	8	4	8
runtime	20hr	40m⋆	35m	5m

F QUALITATIVE ANALYSIS ON THE CLUSTERING RESULTS.

We qualitatively evaluate the quality of our clusters both on the experimental structures in PDB and the AlphaFold structures in Uniclust30. First, we measured how many unique token IDs occurred in each protein in Figure 4a. We observed that over 20% of the proteins contained the same token ID for every residue in the sequence. We then measured the number of times each token appeared in the entire Uniclust30 dataset and found that one token appeared over 20% in total (see Figure 4b). This turns out to be token `[17]` in Figure 5 which contains disordered regions with little or no secondary or tertiary structures. Interestingly, the microenvironments in PDB with token `[17]` do contain more sparse environments. This motivated us to remove training on the special token $s^* =$`[17]`.

We also looked at a few tokens in Figure 5 that either occurred the most/least and report our intuition below. Note that while our intuition can offer some rationale about the clusters, the model may capture relevant microenvironment features that are difficult for humans to interpret.

- `[id:3]`: In PDB proteins, this cluster captures semi-solvent exposed microenvironments with masked alanines. In Alphafold proteins, the cluster still contains semi-solvent exposed microenvironments, but not necessarily with a masked alanine. This is the least frequently seen structure token in Uniclust30.
- `[id:14]`: In PDB proteins, this cluster captures solvent-exposed microenvironments with masked glycines. In Alphafold proteins, they correspond to surface exposed with often masked glycines, but also the first amino acid or one in an unfolded loop. Both PDB and Alphafold microenvironments lack structural context. This is the second most frequently seen structure token in Uniclust30. It is the most frequently seen token ID in PDB.
- `[id:17]` In PDB proteins, this cluster contains surface-exposed residues. In Alphafold proteins, this cluster corresponds to unwound proteins without any secondary or tertiary interactions. This is the most frequent structure token in Uniclust30 and the second least frequent structure token in PDB.
- `[id:25]`: In PDB proteins, this cluster contains a lot of cysteine in disulfide bridges. In Alphafold proteins, they correspond to glycine. This is the least frequently seen structure token in PDB.



(a) **Histogram of Unique Structure Token IDs per Protein**

(b) **Distribution of tokens across the entire dataset**

Figure 4: **Measuring the diversity of tokens in both PDB and Uniclust30.**

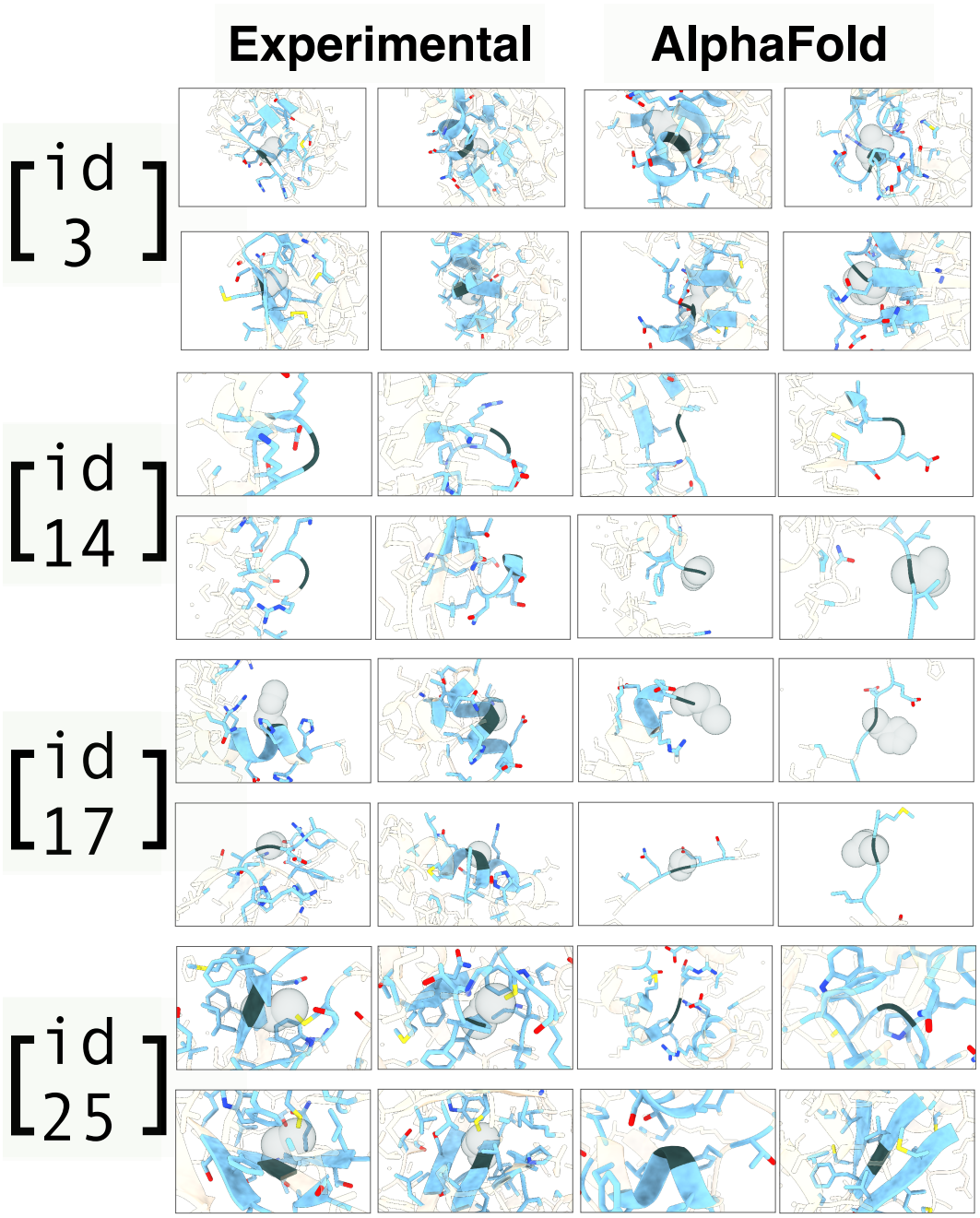


Figure 5: **Cluster Visualizations** of clusters 3, 14, 17, 25. Left two columns are from the PDB, right two columns are from protein sequences UC-30, folded via AlphaFold.

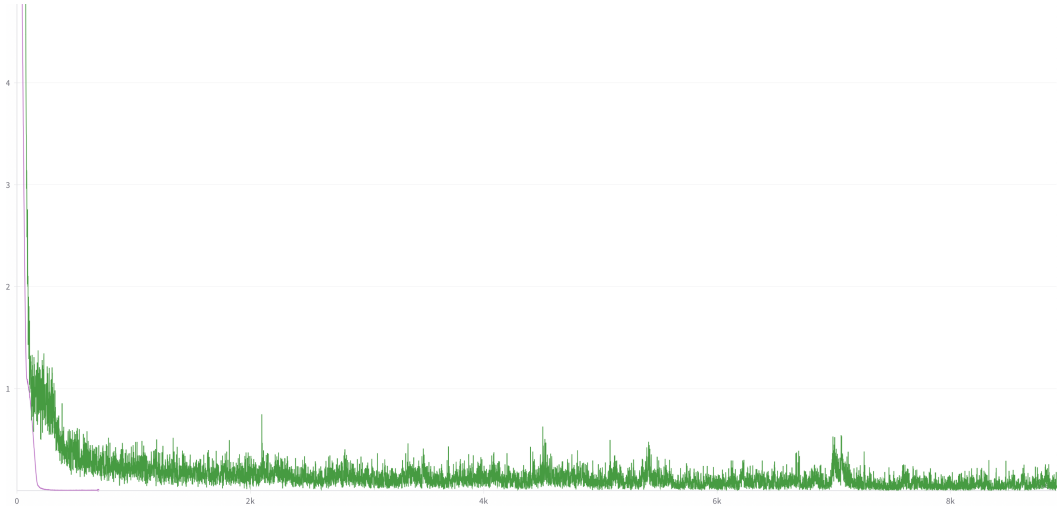


Figure 6: **Autoencoder Training curves.** We plot the coordinate reconstruction loss. Green: Atomic Autoencoder which takes the distances in the bias of the attenion weights. Purple: trivial "autoencoder" which also takes the atomic coordinates directly as input. Atomic Autoencoder trains much slower and needs to extract information about the structure to solve the task while the trivial "autoencoder" does not.