LIGHTWEIGHT EQUIVARIANT GRAPH REPRESENTATION LEARNING FOR PROTEIN ENGINEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

This work tackles the issue of directed evolution in computational protein design that makes accurate predictions of the function of a protein mutant. We design a lightweight zero-shot graph neural network model for multi-task protein representation learning from its 3D structure. Rather than reconstructing and optimizing the protein structure, the trained model recovers the amino acid types and key properties of the central residues from a given noisy three-dimensional local environment. On the prediction of higher-order mutations where multiple amino acid sites of the protein are mutated simultaneously, the proposed strategy achieves remarkably higher performance by 20% improvement at the cost of requiring less than 1% of computational resources that are required by popular transformer-based state-of-the-art deep learning models for protein design.

1 INTRODUCTION

Mutation is a biological process where the amino acid (AA) type of one or multiple sites of a specific protein is changed. While the wild-type proteins' functions do not always meet the demand of bio-engineering, it is vital to manually optimize the functionality, namely fitness, with favorable mutations so that they are applicable in designing antibodies (Wu et al., 2019; Pinheiro et al., 2021; Shan et al., 2022) or enzymes (Sato & Ishida, 2019; Wittmann et al., 2021).

A protein usually constitutes hundreds to thousands of AAs, where each residue belongs to one of twenty AA types. To optimize a protein's functional fitness, a greedy search is usually conducted in the local sequence, where AA sites are mutated to proper AA types to render a protein mutant with the highest gain-of-function (Rocklin et al., 2017). Such a process is called *directed evolution* Arnold (1998). To obtain a mutant with great fitness, multiple AA sites ($\sim$5-10) of the protein need to be mutated, namely *deep mutations* (see Figure 1). It, however, requires enormous experimental costs, as the total number of potential combinations of mutations for deep mutants is astronomical.

Since it is impossible to conduct systematic experimental tests on all possible deep mutations, *in silico* examination of protein variants' fitness becomes highly desirable. A handful of deep learning methods have been developed to accelerate the discovery of advantageous mutants. For instance, Lu et al. (2022) applied 3DCNN to identify a new polymerase with advantageous single-site mutation and enhanced the speed of degrading PET, i.e., a type of solid waste, by 7-8 times at 50°C. Luo et al. (2021) proposed ECNET that predicts functional fitness for protein engineering with evolutionary context. The model guides the engineering of TEM-1 β-lactamase and identifies variants with improved ampicillin resistance. Thean et al. (2022) enhanced SVD with deep learning to predict nuclease variants' activities in multi-site-saturated mutagenesis libraries from and identified Cas9 nuclease variants that possess higher editing activity of derived base editors in human cells.

Due to the scarcity of labeled protein data, researchers often pre-train an encoder for unsupervised learning with protein sequences or structures, and use the learned protein representations to train specific tasks, such as *de novo* protein design (Hsu et al., 2022), mutation effect prediction (Ingraham et al., 2019; Jing et al., 2020; Meier et al., 2021; Notin et al., 2022), and higher-level structure prediction (Elnaggar et al., 2021). In the context of fitness prediction of mutation effect, existing methods usually transform the problem to mini-*de novo* design, which infers a specific AA type from its microenvironment, or analogously its neighboring AA types. Current state-of-the-art sequence-based protein learning methods rely heavily on multiple sequence alignment (MSA; Riesselman et al. (2018); Frazer et al. (2021); Rao et al. (2021)) and protein language models (Elnaggar et al.,

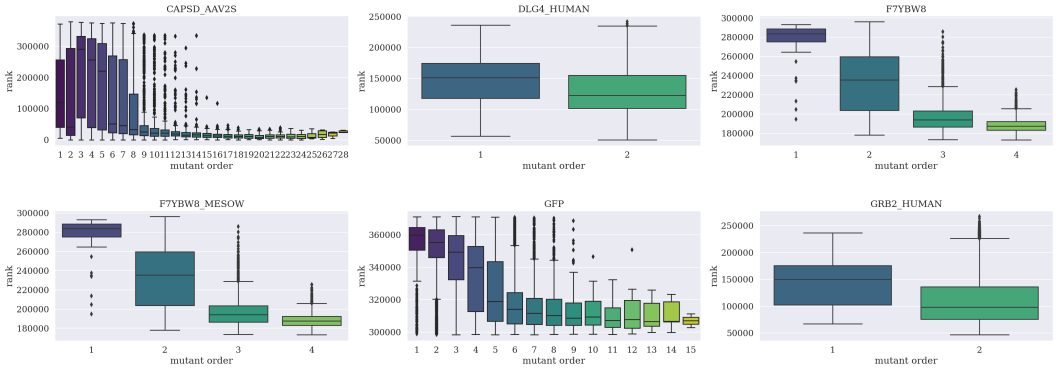

Figure 1: Mutating on more sites frequently results in a higher score, i.e., a smaller rank value.

2021; Rives et al., 2021; Nijkamp et al., 2022; Brandes et al., 2022). While MSA helps capture important evolutionary properties of the protein family, it nevertheless multiplies the requirements of computing resources. The latter protein language models derived from natural language processing (NLP) encode sequence semantics and often need hundreds of GPU cards to train on hundreds of millions of proteins. Meanwhile, an autoregressive inference process is usually required along the entire protein sequence to score a mutation on a single site, which further slows down the inference speed (Sato & Ishida, 2019; Liu et al., 2022; Hsu et al., 2022; Notin et al., 2022). More importantly, when predicting the fitness of the higher-order mutants, most of these models made a crude assumption that the mutations on different sites happen sequentially or individually, which is incorrect in most cases (Lehner, 2011; Breen et al., 2012). The ignored epistatic effects between different sites are potentially a key factor hindering the acquisition of favorable high-order mutants in directed evolution (Sarkisyan et al., 2016; Rollins et al., 2019).

Mutation of AA sites also occurs in nature, where an AA site might be mutated to any of the other 19 AA types in a random manner. It is suggested by natural selection that only the mutants that exhibit the best fitness and fit the environment survive. As a protein's functionality is determined by its structure, we encode the folded protein by a *protein graph* with AAs being graph nodes to provide an elegant 3D spatial description of the protein. The first-level information, such as AA types, spatial coordinates of Cα, and C-N angles between neighboring AAs, are embedded in node features. Altering AA types of a protein in nature can then be viewed as adding corruptions to the node features of the protein graph, and denoising the graph makes a remedy to search for mutants with the best fitness. We model the protein mutation effect prediction as a denoising problem with equivariant graph neural networks (Satorras et al., 2021). For a given protein, the recovered predictions can be leveraged to forecast the fitness of deep mutational effects and discover favorable mutants.

Compared to existing state-of-the-art deep learning methods for mutation effect prediction, such as ESM-1v (Meier et al., 2021) and ESM-IF1 (Hsu et al., 2022), the designed lightweight equivariant graph neural network (LGN) stands out in three perspectives.

First, **LGN improves generalization ability** through the multi-task learning strategy and biological prior knowledge. The pre-trained model encodes the chemical and physical properties of a given AA's microenvironment with domain knowledge for practically meaningful representations.

Secondly, **LGN avoids the independent-mutation assumptions** by generating the probabilities of all the amino acid residues at a time, which implements the joint distribution of all variations. In literature, the higher-order mutation effect is usually approached by summing up log-odd-ratio scores of the corresponding individual single-site mutants. The linear combination over separately assigned predictions is unsubstantiated, as the independent mutations neglect the epistatic effect.

Thirdly, **LGN is efficient in both the training and inference phases**. The spatial graph inputs portray the topological properties of proteins, which circumvents data augmentation that is typically required by sequence or grid representations. Equivariant message passing, alternatively, provides a feature distillation unit with translation and rotation equivariance and encodes AAs' microenvironment defined by the protein graph's geometry.

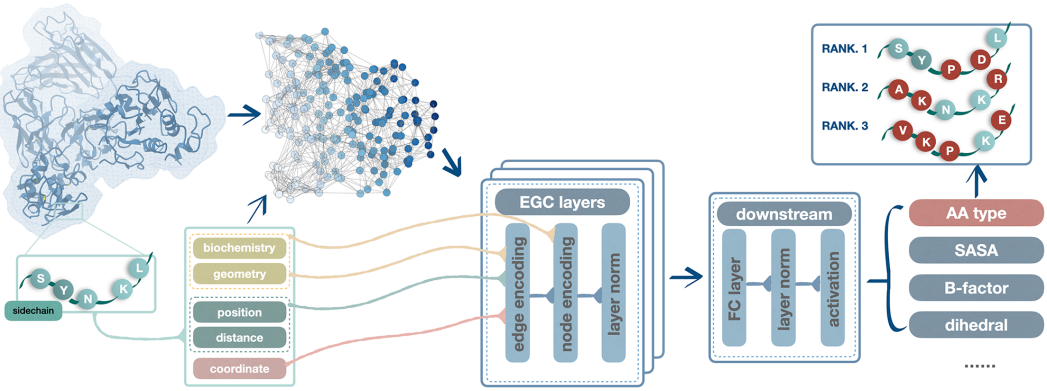

Figure 2: An illustration of the proposed LGN. The model is pre-trained with a set of protein graphs that are featured by (perturbed) node attributes and 3D positions with a multi-task learning strategy. A stack of EGC layers encodes rotation and translation equivariant structural representations for each node on individual graphs. Next, fully-connected layers are employed to learn different labels, where the AA type prediction is used for suggesting top-ranked mutations.

2 Zero-shot Learning for Protein Recovery

The excessive cost in laboratory results in scarce mutation scanning data, especially deep mutant results. It is thus favorable to pre-train a zero-shot protein prediction model that can be generalized directly to an unseen task without any further supervision to specialize the model.

2.1 Graph Representation of Protein Structure

For a given protein, we create a k-nearest neighbor (kNN) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to describe its 3D structure and molecular properties. Here each node $v_i \in \mathcal{V}$ represents an amino acid residue with $\boldsymbol{X} \in \mathbb{R}^{34}$ node attributes constituting biochemical properties and geometric properties of amino acids. The former includes 20-dimensional attributes of one-hot encoded amino acid types ($\boldsymbol{X}_{\text{aa}}$), two scalars for each residue, i.e., solvent-accessible surface area (SASA) and the standardized crystallographic B-factor, and 5 normalized surface-aware node features. The geometric properties include the direction position ($\boldsymbol{X}_{\text{pos}}$) of each residue by 3D coordinates of its α-carbon and the relative position of the amino acid in the protein chain ($\boldsymbol{X}_{\text{agl}}$) by the dihedral angles $\{\sin, \cos\} \circ \{\phi, \psi\}$ computed from the backbone atom positions. For a specific v_i of the ith amino acid in the protein sequence, the dihedral angles are measured from $C\alpha_{i-1}, N_i, C\alpha_i, N_{i+1}$.

To build edge connections, we first define a symmetric adjacency matrix $\boldsymbol{A}$ with the kNN-graph to capture the nodes' microenvironment, i.e., each node is connected to up to k other nodes in the graph that has the smallest Euclidean distance over other nodes, and the distance is smaller than a certain cutoff (30$\mathring{A}$). Consequently, if v_i and v_j are connected to each other, we have $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji} \neq 0$. The edge attributes $\boldsymbol{E} \in \mathbb{R}^{93}$ feature the connected edges in $\mathcal{E}$, including 15 inter-atomic distances, 12 local N-C positions, and the relative position in the protein sequence in 66-dimensions.

2.2 Pre-training with Prior Domain Knowledge for Better Protein Fitness

The wild-type proteins suffer from random perturbations or mutations that not every AA site has the best AA type (Liu et al., 2022). To this end, we pre-train our model with a multitask learning strategy, which removes the natural corruptions and predicts key protein properties to help encode the microenvironment of the stabilized proteins of interest.

AA Type Denoising We refine $\boldsymbol{x}_{\text{aa}}$, the AA type a node, to $\tilde{\boldsymbol{x}}_{\text{aa}}$ with a Bernoulli noise, i.e.,

$$\pi(\tilde{\boldsymbol{x}}_{\text{aa}}|\boldsymbol{x}_{\text{aa}}) = p\delta(\tilde{\boldsymbol{x}}_{\text{aa}} - \boldsymbol{x}_{\text{aa}}) + (1-p)\mathcal{M}(n, \pi_1, \pi_2, ..., \pi_n), \tag{1}$$

where the confidence level p is a tunable parameter that controls the proportion of residues that are 'noise-free'. The probability for the residue to become a particular type depends on the distribution

of the 20 types $\mathcal{M}(n, \pi_1, \pi_2, ..., \pi_n)$, which involves prior knowledge in molecular biology. This paper defines the distribution by the observed probability density of amino acid types in wild-type proteins [1]. See Appendix E to better understand the influence of different confidence levels.

Geometric Properties Denoising For the continuous-valued features, such as 3D coordinates and dihedral angles, an i.i.d Gaussian noise is introduced, learning to remove which corresponds to approximating the data-generating force field of molecules (Zaidi et al., 2022). To be specific,

$$\tilde{\boldsymbol{x}}_{\text{pos}} = \boldsymbol{x}_{\text{pos}} + \sigma\epsilon, \quad \text{where } \epsilon \sim \mathcal{N}(0, I_3). \tag{2}$$

The noise effect is determined by σ, which is tunable to fit the scale of the noiseless raw feature.

Bio-chemistry Properties Recovery Aside from denoising the perturbed residues type and geometric properties, other auxiliary tasks are introduced to help establish an expressive hidden microenvironment representation. Specifically, SASA is known to strongly influence AA type preferences, and B-factors are associated with the conformations and mobility of the neighboring AA. We thus introduce inductive biases to the model by predicting these two properties in the output.

Label Smoothing with Amino Acid Substitution Matrices Protein sequence alignments provide important insights for understanding gene and protein functions. The similarity measurement of an alignment of protein sequence reflects the favors of all possible exchanges of one amino acid with another. We employ BLOSUM (Henikoff & Henikoff, 1992), a substitution matrix, to account for the relative substitution frequencies and chemical similarity of AAs. The matrix is derived from the statistics for every conserved region of protein families in **BLOCKS** database. As AA sites are more likely to be mutated to the AA type within the block of high similarity scores in the BLOSUM table, we hereby modify our loss function so that a mutation to an AA type with a higher similarity score accumulates a smaller penalty than to the one with a lower similarity score.

2.3 Protein Structure Representation with Equivariant GNNs

Proteins are structured in the 3-dimensional space, and it is vital for the model to predict the same binding complex no matter how the input proteins are positioned and oriented. Instead of practicing expensive data augmentation strategies, we follow Satorras et al. (2021) and construct SE(3)-equivariant neural layers for graph embedding. At the lth layer, an Equivariant Graph Convolution (EGC) inputs a set of n hidden node properties embedding $\boldsymbol{H}^l = \left\{\boldsymbol{h}_1^l, \ldots, \boldsymbol{h}_n^l\right\}$ as well as the node coordinate embeddings $\boldsymbol{X}_{\text{pos}}^l = \left\{\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_n^l\right\}$ for a graph of n nodes. The attributed edges are denoted as $\boldsymbol{E} = \{\ldots, \boldsymbol{e}_{ij}, \ldots\}$. The target of an EGC layer is to output a transformation on the node feature embedding $\boldsymbol{H}^{l+1}$ and coordinate embedding $\boldsymbol{X}_{\text{pos}}^{l+1}$. Concisely: $\boldsymbol{H}_{\text{pos}}^{l+1}, \boldsymbol{X}^{l+1} = \text{EGC}\left[\boldsymbol{H}^l, \boldsymbol{X}_{\text{pos}}^l, \boldsymbol{E}\right]$. To achieve this, an EGC layer defines

$$\begin{aligned}
\boldsymbol{m}_{ij} &= \phi_e\left(\mathbf{h}_i^l, \mathbf{h}_j^l, \left\|\mathbf{x}_i^l - \mathbf{x}_j^l\right\|^2, \boldsymbol{e}_{ij}\right) \\
\boldsymbol{x}_i^{l+1} &= \mathbf{x}_i^l + \frac{1}{n}\sum_{j \neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right) \\
\boldsymbol{m}_i &= \sum_{j \neq i}\mathbf{m}_{ij} \\
\boldsymbol{h}_i^{l+1} &= \phi_h\left(\mathbf{h}_i^l, \boldsymbol{m}_i\right),
\end{aligned} \tag{3}$$

where ϕ_e, ϕ_h are respectively the edge and node propagation operations, such as multi-layer perceptrons (MLPs). The ϕ_x is an additional operation that projects the vector embedding $\boldsymbol{m}_{ij}$ to a scalar value. The EGC layer preserves equivariance to rotations and translations on the set of 3D node coordinates $\boldsymbol{X}_{\text{pos}}$, while simultaneously performing invariance to permutations on the set of nodes $\mathcal{V}$ in the same fashion as GNNs.

[1] Retrieved from the folded protein dataset by **AlphaFold2** (Varadi et al., 2022) at `https://alphafold.ebi.ac.uk/`

2.4 MODEL OVERVIEW

Our model is depicted in Figure 2. We take a set of protein graphs with attributed nodes and edges, as well as each node's 3D coordinates, as the input to pre-train a zero-shot model. A stack of EGC layers is trained to extract rotation and translation equivariant representations for each node on individual graphs. The hidden representation is then sent to fully-connected layers to establish multiple outputs, such as AA type classification, SASA and B-factor prediction, and 3D coordinates denoising. The total loss for the multitask learning task is given by

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{aa}} + \lambda_1 \mathcal{L}_{\text{sasa}} + \lambda_2 \mathcal{L}_{\text{b-fac}} + \lambda_3 \mathcal{L}_{\text{pos}} + \lambda_4 \mathcal{L}_{\text{agl}}, \tag{4}$$

where $\lambda_i, i = 1, \ldots, 4$ are tunable hyper-parameters to balance different losses on auxiliary regression tasks. These losses are measured by mean squared error (MSE) loss. For AA type classification, we measure its loss $\mathcal{L}_{\text{aa}}$ by cross-entropy with label smoothing technique (Szegedy et al., 2016). The classification loss on an arbitrary node i reads

$$\mathcal{L}_{\text{aa}} = (1-\varepsilon)\Big[-\sum_{y=1}^{20} p(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i) \log q_\theta(\hat{y}_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)\Big] + \varepsilon\Big[-\sum_{y=1}^{20} u(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i) \log q_\theta(\hat{y}_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)\Big],$$

where $p(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ denotes the ground-truth distribution and $q_\theta(\hat{y}_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ is the distribution of predicted labels following a softmax function. In order to improve the generalization and respect the prior biological knowledge, we modify the ground truth label distribution $p(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ from the hard one-hot encoding to $(1-\varepsilon)p(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i) + \varepsilon u(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ when the predicted $\hat{y}_{\text{aa}} = y_{\text{aa}}$ and $\varepsilon u(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ otherwise with some tolerance factor ε. In particular, we define the distribution of $u(y|x_i)$ by the BLOSUM substitution matrix.

3 RESULTS

3.1 EXPERIMENTAL SETUP

We train LGN on **CATH v4.3.0** (Orengo et al., 1997) with artificial noise to predict AA type, 3D coordinates, dihedral angles, and chemical properties (SASA and B-factor). The hidden embeddings of amino acids are learned by SE(3)-equivariant graph convolutions. The performance is validated by a zero-shot prediction task for the fitness of mutation prediction with deep mutational scanning (DMS; Fowler & Fields (2014)) datasets. The model performance is compared against popular state-of-the-art language models and structure-enhanced models.

Baseline Models We compare with a diverse of state-of-the-art models on the fitness of mutation effects prediction. In particular, DEEPSEQUENCE (Riesselman et al., 2018) trains VAE on protein-specific MSAs to capture higher-order interactions from the distribution of an AA sequence. MSA TRANSFORMER Rao et al. (2021) is a language model with aligned protein sequences of interest; ESM-1V (Meier et al., 2021) make zero-shot mutation predictions with masked language modeling; and ESM-IF1 (Hsu et al., 2022) predicts protein sequence with GVP (Jing et al., 2020), a graph representation learning methods for vector and scalar features of protein graphs. Furthermore, both TRANCEPTION (Notin et al., 2022) and PROGEN2 (Nijkamp et al., 2022) leverages autoregressive language models to retrieve AA sequence without family-specific MSAs.

Lightweight Equivariant Graph Neural Networks (LGN) To train our LGN framework, we first generate protein graphs for the sequences in **CATH**. See Appendix A for a detailed introduction to the generation, and Appendix C for a summary of the generated dataset. For the total number of $31,848$ protein graphs of 150 nodes on average, we randomly pick 500 graphs for validation and leave the remaining for model fitting. During the learning phase, we assign random perturbations to AA types and other features we mentioned earlier in Section 2. The noises are fixed to guarantee stable and comparable measurements at the validation step. The specific influence of different choices on the hyper-parameters (e.g., the noise level) will be discussed later in this section and Appendix D-E. The main architecture constitutes a stack of 6 EGC layers following 1 fully-connected layer to make predictions on the different learning tasks. On each node, the output is a vector representation consisting of 20 probabilities of the masked amino acid, 1 predicted SASA, 1 B-factors,

Table 1: Performance Comparison of Baseline Models on DMS assays prediction. Results with a higher Spearman's correlation are preferred.

	DEEPSEQUENCE	TRANCEPTION	PROGEN2	MSA TRANSFORMER	ESM-1V	ESM-IF1	LGN (ours)
CAPSD	**0.4831**	0.2307	0.2112	**0.4419**	0.2212	0.2170	**0.3589**
DLG4_HUMAN		**0.6200**	0.5712	0.4654	0.4654	**0.6164**	**0.6197**
F7YBW8	**0.3939**	**0.4280**	0.3231	0.3483	0.2865	0.3714	**0.4223**
F7YBW8_MESOW	**0.4205**	**0.4036**	0.3231	0.3631	0.2522	0.3690	**0.4210**
GFP	0.6331	**0.6647**	0.6459	**0.7069**	**0.7208**	0.6203	0.6455
GRB2_HUMAN	0.3886	0.4441	**0.5211**	0.2808	0.3211	**0.7033**	**0.6044**
average correlation	**0.4698**	0.4177	0.4186	0.4511	0.3756	**0.4714**	**0.5071**

† The top three are highlighted by **First**, **Second**, **Third**.

and 4 dihedral values (when applicable). The 3D coordinates are derived directly from EGC outputs. The loss function by Equation 4 guides the backward propagation with ADAM (Kingma & Ba, 2015) optimizer. The model is trained with 300 epochs with the initial rate set to 0.001 and weight decay to 0.01. The learning rate is dampened to 0.0001 after 150 epochs.

Evaluation All the models are evaluated with deep mutational scanning (DMS) assays that assess a diverse set of 15 proteins, where 9 of them only contains single-site mutation scores, and 6 of them have both single-site and higher-order mutational records (see Appendix B). The protein structures are folded from the provided sequence information with ALPHAFOLD2 (Jumper et al., 2021), following the exact same pre-processing steps as in **CATH** for generating protein graphs. The only difference is that we do not append artificial noises onto the test proteins, as we assume they are already noisy. We then send the unmutated test proteins to the pre-trained LGN model and use the log-odd-ratio in Equation 5 of the predicted probabilities of AA types for suggesting the rank of deep mutations. The prediction performance is evaluated on Spearman's correlation coefficient between the computational and experimental scores on all the mutation combinations.

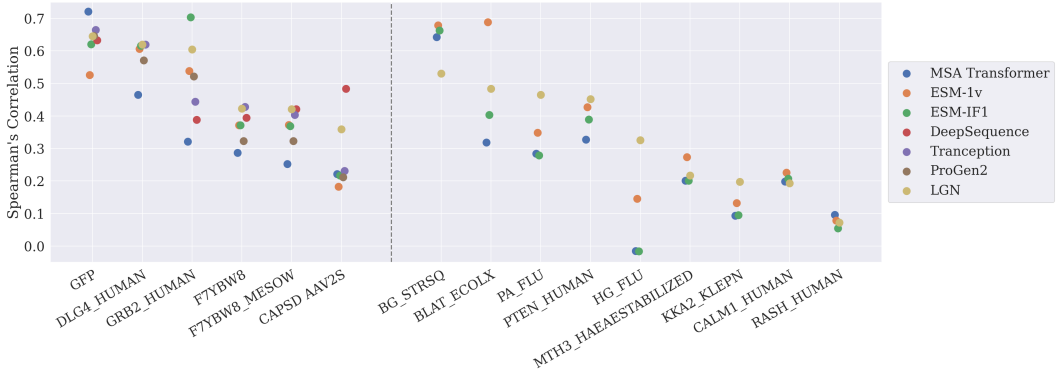

Figure 3: Per task performance on the fitness of deep mutant prediction with pre-trained models. Each point indicates Spearman's correlation coefficients on the corresponding protein. The left 6 proteins contain higher-order mutations, and the right 9 proteins record shallow mutants.

3.2 FITNESS OF DEEP MUTANTS PREDICTION

The first experiment evaluates the fitness of proteins' mutation effects prediction, where the fitness scores are inferred directly from a pre-trained model without supervision on a task-specific model. We visualize the overall performance comparison on protein-wise Spearman's correlation coefficients in Figure 3. The deep mutation scores are reported in Table 1, where LGN outperforms baseline methods and achieves at least comparable results in single-site mutant tests. Overall, our model achieves 0.5071 weighted average correlations on deep mutant effect predictions. While DEEPSEQUENCE achieves superior performance over the majority rest, it should be noticed that the model has to be trained on every new protein, and it cannot be generated for other proteins. Also, the training speed and performance of the learned model rely heavily on the quality of the available MSA information, which can very a lot on different proteins.

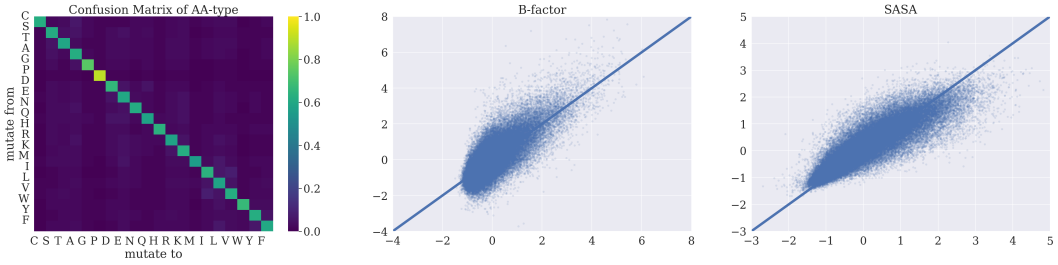

Figure 4: The three plots (from left to right) display the confusion matrix of predicted AA types, and linear regression on the predicted SASA and B-factor, respectively.

3.3 Protein Recovery

This experiment investigates the auxiliary learning tasks of the pre-trained model, including the learning performance in predicting AA types, SASA, and B-factor. In specific, we visualize the confusion matrix of the predicted AA types with respect to the ground-truth AA types to see evaluate the model's capability to recover from noisy sequences to the original sequence, i.e., if the large values are accumulated to the diagonal of the confusion matrix. For the SASA and B-factor predictions, we examine the R^2 of the predicted and the ground-truth values on **CATH**. The results are visualized in Figure 4 with denoised AA type, as well as the predicted SASA and B-factor as the output tasks. The AA type prediction achieves high accuracy with the majority of predictions accumulated on the diagonal line. For the two regression tasks, we fit the true value and the predicted value with linear regression. The estimated coefficients are 1.008 and 0.989 for B-factor and SASA, respectively. The p-value for both coefficients is < 0.001. In addition, Pearson's correlation coefficients for the two variants are respectively 0.884 and 0.791.

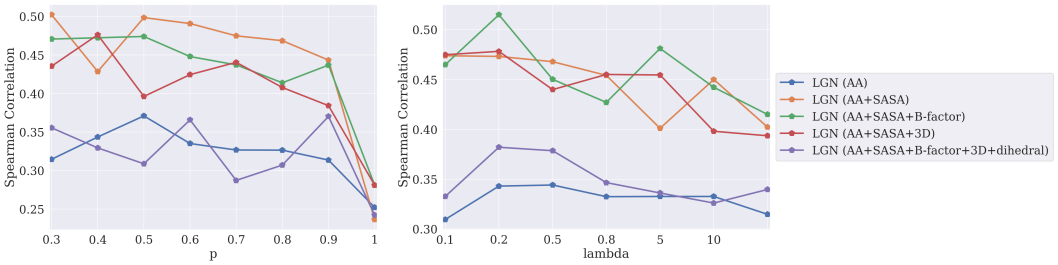

Figure 5: Average Performance with different ps (left) and λs (right) on the auxiliary tasks.

3.4 Selection Criteria on the Additional Hyper-parameters

As LGN introduces two additional hyper-parameters in training the model, this section examines the influence of selecting different ps and λs. Figure 5 visualizes a selection of model performance on the Spearman's correlation with variant ps and λs with wild-type noise distribution.

While the choice of p can be determined by prior knowledge regarding the quality of wild-type proteins, here we treat p as a data-driven hyper-parameter to be optimized during the model training. We exclude extremely small ps to avoid drastic perturbation rates and search for the optimal $p \in \{0.3, 0.4, \ldots, 0.9, 1\}$. The different choices on ps are validated with different learning tasks on the left side of Figure 5 for higher-order mutants. In general, a moderate p between 0.3 and 0.6 best suits the majority selection of learning modules and noise distribution. Based on the overall performance, we suggest $p = 0.6$ as the default value of the confidence level (See Appendix E for more results on different types of perturbation noise and proteins).

We also investigate a wide range of the choices of λs. For simplicity, we let $\lambda_1 = \lambda_2 = \lambda_3 \in \{0.05, 0.1, 0.2, 0.5, 0.8, 5, 10\}$ and fix $\lambda_4 = 0.5$. All the results are conducted under the recommended $p = 0.6$ with wild-type noise. We report the average performance on deep mutants in the

Table 2: Comparison of baseline models. The train and inference speed is tested on **GFP**.

model	DEEPSEQUENCE	MSA TRANS.	ESM-1V	ESM-IF1	TRANCEPTION	PROGEN2	LGN (ours)
input	sequence	sequence	sequence	sequence+structure	sequence	sequence	structure
MSA	✓	✓			✓		
train on new protein	✓				✓		
training dataset	-	**Uniref50**	**Uniref90**	**CATH+AF2**	**Uniref100**	**Uniref90+BFD30**	**CATH**
	-	(2018-03)		(2020-03)			v4.3.0
training size (M)	-	45	98	12	249	$> 1,000$	0.03
max. input token		$1,024$	$1,024$	$1,024$	$1,280$	$1,024$	$2,687$[1]
# parameters (M)	4.3	100	650	142	700	$2,700$	1.5
# layers	$1,600$	12	-	20	36	32	6
# head	-	12	-	8	20	32	-
# hid. dim.	$100 - 2,000$	-	-	$512 - 2,048$	-	-	512
speed (training day)	-	13[2]	6	653	$\sim$100	-	0.17
resource (train)	-	128×V100[2]	64×V100	32×V100	64×A100	?×TPU-v3	1×3090
preparing speed (sec)	$6,360 + 25,020$	$6,360$	-	-	$6,360$	-	-
inference speed (sec)	608	927	75	102	$1,920$	$1,440$	25

right plot of Figure 5, which demonstrates a relatively flat and steady trend with a mild peak at $\lambda = 0.2, 0.5$. Additional results are provided in Table 8 of Appendix E with various model setups.

3.5 INFERENCE SPEED

LGN consumes significantly fewer computational resources in training and inference. We compare the model scale, inference time, and prediction performance in Figure 6 and Table 2.

The model size and the required resources with the baseline methods are provided by the authors. As the majority of models are pre-trained, we record the inference speed on a single 3090 GPU. While the time cost is significantly lower than experimental methods, we measure it to indicate the cost of forward propagation in one iteration, which can be viewed as the indirect empirical evidence of the training cost. As each protein requires independent inference progress, we hereby take **GFP** as an example protein sample. The protein constitutes 236 amino acid residues, and it has over $50,000$ mutant records (see Table 3 in Appendix B for more details). Note that: 1). The $2,687$ input token length only refers to the maximum protein length we used during training. In fact, the model itself can process large protein graphs containing over tens of thousands of amino acids. 2). The training speed and required resources for MSA TRANSFORMER are retrieved from Meier et al. (2021). The original work by Rao et al. (2021) only reports that they used 32×V100 GPUs for training, without revealing the training time.

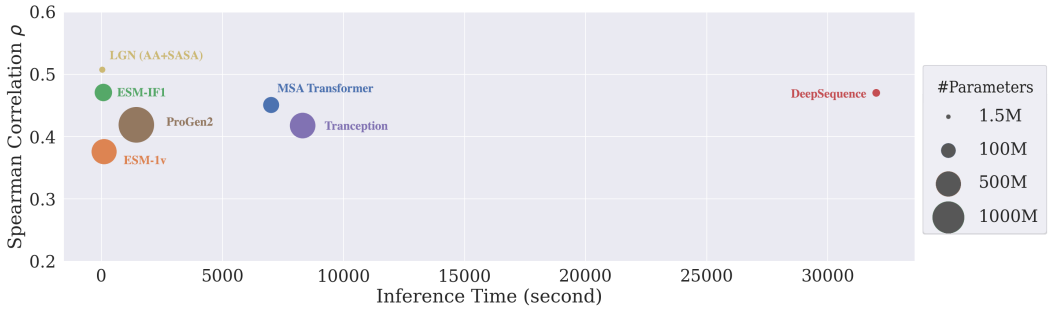

Figure 6: Comparison of Inference Efficiency. The area of the ball indicates the number of network parameters of a model. Our model (in blue) can achieve SOTA performance (y-axis) with minimum inference time (x-axis) and 1% number of parameters of the ESM.

4 RELATED WORK

Protein Sequence and Structure Representation Due to the enormous experimental cost of measuring protein structures, the number of known protein sequences is thousands of times larger than

protein structures (Eswar et al., 2008; Hsu et al., 2022). Meanwhile, the protein sequences representation is highly similar to human language, which naturally promotes the fast development of natural language processing (NLP), especially transformer-based methods for encoding protein sequences (Coin et al., 2003; Meier et al., 2021; Ofer et al., 2021; Castro et al., 2022). However, the geometry of proteins also suggests higher-level structures and topological relationships that are vital to protein functionality. Structure prediction of proteins always attracts great attention in the field (Chi & Liberles, 2016; Jumper et al., 2021; Baek et al., 2021; Varadi et al., 2022). The breakthrough progress in protein folding also enriches structured proteins. For instance, Hsu et al. (2022) and Ma et al. (2022) mixed experimentally-tested and ALPHAFOLD-predicted for model training, which greatly eases the data shortage problem and achieves significant performance gain.

Structural encoding for Protein Graphs According to the laws of physics, the atomic dynamics do not change no matter how a protein is translated or rotated from one place to another (Han et al., 2022). Therefore, the inductive bias of symmetry should be incorporated into the design of protein structure-based models. To this end, research work has been proposed to respect the spatial relationship of amino acids (Torng & Altman, 2017; Sato & Ishida, 2019). Such CNN-based methods aggregate the local structure of each residue and integrate estimated local qualities into the whole protein properties. However, these methods neglect geometric equivariance, which can usually be captured by equivariant graph neural networks (Ganea et al., 2021; Stärk et al., 2022).

Protein Representation As existing protein language models require high computational costs and are difficult to train, finding an effective feature representation of protein data is important for downstream tasks (Thompson et al., 2012). Contrastive learning and self-prediction (Elnaggar et al., 2021; Zhang et al., 2022; Hsu et al., 2022) used self-supervised pre-training methods to extract good representation for reducing computational resources. Despite only applying classical representation learning methods on protein, some researchers designed sophisticated encoders for expressive protein representation. For instance, Li et al. (2022) proposed $\vec{W}$-GNN variants that efficiently interact with scalar-vector features. Somnath et al. (2021) introduced HOLOPROT to connect different modalities of proteins, including surface, structure, and sequence representation.

Mutation Effect Prediction Multiple sequence alignment (MSA) is an essential ingredient for many of the existing state-of-the-art methods to predict the effect of single amino acid substitutions such as DEEPSEQUENCE (Riesselman et al., 2018), ALPHAFOLD2 (Jumper et al., 2021), MSA TRANSFORMER (Rao et al., 2021), and LM-GVP (Wang et al., 2022). The MSA for a protein sequence or domain captures meaningful information on the evolutionary information of the protein within its family at the cost of bringing severe limitations–not all proteins are alignable, such as CDRs of antibody variable domains (Shin et al., 2021), and not all the alignments are deep enough to train models sufficiently large to learn the complex interactions between residues. To deal with this issue, ESM-1V (Meier et al., 2021) trains a zero-shot model on a large set of unaligned sequences to secure a scalable and bias-free training procedure, and TRANCEPTION (Notin et al., 2022) leverages autoregressive predictions and retrieval of homologous sequences at inference.

5 CONCLUSION

Designing directed evolution on proteins, especially with deep mutants for functional fitness, is of enormous engineering and pharmaceutical importance. However, existing experimental methods are economically costly, and *in silico* methods require significant computational resources. This paper proposed a lightweight zero-shot model for mutant effect prediction on arbitrary numbers of AAs by transferring the problem to denoising a protein graph. Our model is trained to recover AA types and other important properties (e.g., B-factor, SASA, and the spatial position of Cα) from observed noisy proteins. We employ translation and rotation equivariant neural message passing layers to extract geometric-aware representation for the microenvironment of central AAs and thus grasp rich information for efficiently learning protein function. The model achieves state-of-the-art performance on PDB datasets in deep mutant tests with significantly fewer computational resources than existing SOTA models.

REFERENCES

Frances H Arnold. Design by directed evolution. *Accounts of chemical research*, 31(3):125–131, 1998.

Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

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.

Michael S Breen, Carsten Kemena, Peter K Vlasov, Cedric Notredame, and Fyodor A Kondrashov. Epistasis as the primary factor in molecular evolution. *Nature*, 490(7421):535–538, 2012.

Egbert Castro, Abhinav Godavarthi, Julian Rubinfien, Kevin Givechian, Dhananjay Bhaskar, and Smita Krishnaswamy. Transformer-based protein generation with regularized latent space optimization. *Nature Machine Intelligence*, 4(10):840–851, 2022.

Peter B Chi and David A Liberles. Selection on protein structure, interaction, and sequence. *Protein Science*, 25(7):1168–1178, 2016.

Lachlan Coin, Alex Bateman, and Richard Durbin. Enhanced protein domain discovery by using language modeling techniques from speech recognition. *Proceedings of the National Academy of Sciences*, 100(8):4516–4520, 2003.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Wang Yu, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Towards cracking the language of lifes code through self-supervised deep learning and high performance computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021. doi: 10.1109/TPAMI.2021.3095381.

Narayanan Eswar, David Eramian, Ben Webb, Min-Yi Shen, and Andrej Sali. Protein structure modeling with modeller. In *Structural proteomics*, pp. 145–159. Springer, 2008.

Douglas M Fowler and Stanley Fields. Deep mutational scanning: a new style of protein science. *Nature methods*, 11(8):801–807, 2014.

Jonathan Frazer, Pascal Notin, Mafalda Dias, Aidan Gomez, Joseph K Min, Kelly Brock, Yarin Gal, and Debora S Marks. Disease variant prediction with deep generative models of evolutionary data. *Nature*, 599(7883):91–95, 2021.

Octavian-Eugen Ganea, Xinyuan Huang, Charlotte Bunne, Yatao Bian, Regina Barzilay, Tommi S Jaakkola, and Andreas Krause. Independent se (3)-equivariant models for end-to-end rigid protein docking. In *International Conference on Learning Representations*, 2021.

Jiaqi Han, Yu Rong, Tingyang Xu, and Wenbing Huang. Geometrically equivariant graph neural networks: A survey. *arXiv:2202.07230*, 2022.

Rhys Heffernan, Kuldip Paliwal, James Lyons, Abdollah Dehzangi, Alok Sharma, Jihua Wang, Abdul Sattar, Yuedong Yang, and Yaoqi Zhou. Improving prediction of secondary structure, local backbone angles and solvent accessible surface area of proteins by iterative deep learning. *Scientific reports*, 5(1):1–11, 2015.

Steven Henikoff and Jorja G Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8946–8970. PMLR, 17–23 Jul 2022.

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

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael John Lamarre Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*, 2020.

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

Diederik P Kingma and Jimmy Ba. ADAM: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representation (ICLR)*, 2015.

Ben Lehner. Molecular mechanisms of epistasis within and between genes. *Trends in Genetics*, 27 (8):323–331, 2011.

Jiahan Li, Shitong Luo, Congyue Deng, Chaoran Cheng, Jiaqi Guan, Leonidas Guibas, Jian Peng, and Jianzhu Ma. Directed weight neural networks for protein structure representation learning. *arXiv:2201.13299*, 2022.

Yufeng Liu, Lu Zhang, Weilun Wang, Min Zhu, Chenchen Wang, Fudong Li, Jiahai Zhang, Houqiang Li, Quan Chen, and Haiyan Liu. Rotamer-free protein sequence design based on deep learning and self-consistency. *Nature Computational Science*, 2022.

Hongyuan Lu, Daniel J Diaz, Natalie J Czarnecki, Congzhi Zhu, Wantae Kim, Raghav Shroff, Daniel J Acosta, Bradley R Alexander, Hannah O Cole, Yan Zhang, et al. Machine learning-aided engineering of hydrolases for pet depolymerization. *Nature*, 604(7907):662–667, 2022.

Yunan Luo, Guangde Jiang, Tianhao Yu, Yang Liu, Lam Vo, Hantian Ding, Yufeng Su, Wesley Wei Qian, Huimin Zhao, and Jian Peng. Ecnet is an evolutionary context-integrated deep learning framework for protein engineering. *Nature communications*, 12(1):1–14, 2021.

Wenjian Ma, Shugang Zhang, Zhen Li, Mingjian Jiang, Shuang Wang, Weigang Lu, Xiangpeng Bi, Huasen Jiang, Henggui Zhang, and Zhiqiang Wei. Enhancing protein function prediction performance by utilizing alphafold-predicted protein structures. *Journal of Chemical Information and Modeling*, 2022.

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29287–29303, 2021.

Erik Nijkamp, Jeffrey Ruffolo, Eli N Weinstein, Nikhil Naik, and Ali Madani. Progen2: exploring the boundaries of protein language models. *arXiv:2206.13517*, 2022.

Pascal Notin, Mafalda Dias, Jonathan Frazer, Javier Marchena Hurtado, Aidan N Gomez, Debora Marks, and Yarin Gal. Tranception: Protein fitness prediction with autoregressive transformers and inference-time retrieval. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16990–17017. PMLR, 17–23 Jul 2022.

Dan Ofer, Nadav Brandes, and Michal Linial. The language of proteins: Nlp, machine learning & protein sequences. *Computational and Structural Biotechnology Journal*, 19:1750–1758, 2021.

CA Orengo, AD Michie, S Jones, DT Jones, MB Swindells, and JM Thornton. Cath – a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997. ISSN 0969-2126. doi: https://doi.org/10.1016/S0969-2126(97)00260-8.

Fernanda Pinheiro, Omar Warsi, Dan I Andersson, and Michael Lässig. Metabolic fitness landscapes predict the evolution of antibiotic resistance. *Nature Ecology & Evolution*, 5(5):677–687, 2021.

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

Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature methods*, 15(10):816–822, 2018.

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

Gabriel J Rocklin, Tamuka M Chidyausiku, Inna Goreshnik, Alex Ford, Scott Houliston, Alexander Lemak, Lauren Carter, Rashmi Ravichandran, Vikram K Mulligan, Aaron Chevalier, et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357 (6347):168–175, 2017.

Nathan J Rollins, Kelly P Brock, Frank J Poelwijk, Michael A Stiffler, Nicholas P Gauthier, Chris Sander, and Debora S Marks. Inferring protein 3d structure from deep mutation scans. *Nature genetics*, 51(7):1170–1176, 2019.

Karen S Sarkisyan, Dmitry A Bolotin, Margarita V Meer, Dinara R Usmanova, Alexander S Mishin, George V Sharonov, Dmitry N Ivankov, Nina G Bozhanova, Mikhail S Baranov, Onuralp Soylemez, et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, 2016.

Rin Sato and Takashi Ishida. Protein model accuracy estimation based on local structure quality assessment using 3d convolutional neural network. *PloS one*, 14(9):e0221347, 2019.

Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In *International conference on machine learning*, pp. 9323–9332. PMLR, 2021.

Sisi Shan, Shitong Luo, Ziqing Yang, Junxian Hong, Yufeng Su, Fan Ding, Lili Fu, Chenyu Li, Peng Chen, Jianzhu Ma, et al. Deep learning guided optimization of human antibody against sars-cov-2 variants with broad neutralization. *Proceedings of the National Academy of Sciences*, 119(11): e2122954119, 2022.

Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature communications*, 12(1):1–11, 2021.

Vignesh Ram Somnath, Charlotte Bunne, and Andreas Krause. Multi-scale representation learning on proteins. *Advances in Neural Information Processing Systems*, 34:25244–25255, 2021.

Hannes Stärk, Octavian Ganea, Lagnajit Pattanaik, Regina Barzilay, and Tommi Jaakkola. Equibind: Geometric deep learning for drug binding structure prediction. In *International Conference on Machine Learning*, pp. 20503–20521. PMLR, 2022.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Dawn GL Thean, Hoi Yee Chu, John HC Fong, Becky KC Chan, Peng Zhou, Cynthia Kwok, Yee Man Chan, Silvia YL Mak, Gigi CG Choi, Joshua WK Ho, et al. Machine learning-coupled combinatorial mutagenesis enables resource-efficient engineering of crispr-cas9 genome editor activities. *Nature Communications*, 13(1):1–14, 2022.

Andrea D Thompson, Amanda Dugan, Jason E Gestwicki, and Anna K Mapp. Fine-tuning multiprotein complexes using small molecules. *ACS chemical biology*, 7(8):1311–1320, 2012.

Wen Torng and Russ B Altman. 3d deep convolutional neural networks for amino acid environment similarity analysis. *BMC bioinformatics*, 18(1):1–23, 2017.

Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 50(D1):D439–D444, 2022.

Zichen Wang, Steven A Combs, Ryan Brand, Miguel Romero Calvo, Panpan Xu, George Price, Nataliya Golovach, Emmanuel O Salawu, Colby J Wise, Sri Priya Ponnapalli, et al. Lm-gvp: an extensible sequence and structure informed deep learning framework for protein property prediction. *Scientific reports*, 12(1):1–12, 2022.

Bruce J Wittmann, Kadina E Johnston, Zachary Wu, and Frances H Arnold. Advances in machine learning for directed evolution. *Current opinion in structural biology*, 69:11–18, 2021.

Zachary Wu, SB Jennifer Kan, Russell D Lewis, Bruce J Wittmann, and Frances H Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

Sheheryar Zaidi, Michael Schaarschmidt, James Martens, Hyunjik Kim, Yee Whye Teh, Alvaro Sanchez-Gonzalez, Peter Battaglia, Razvan Pascanu, and Jonathan Godwin. Pre-training via denoising for molecular property prediction. *arXiv:2206.00133*, 2022.

Zuobai Zhang, Minghao Xu, Arian Jamasb, Vijil Chenthamarakshan, Aurelie Lozano, Payel Das, and Jian Tang. Protein representation learning by geometric structure pretraining. *arXiv:2203.06125*, 2022.

A FROM PROTEIN TO GRAPH REPRESENTATION

A.1 GRAPH REPRESENTATION

For a given protein, we create a kNN-Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to describe its 3D structure and molecular properties. Here each node $v_i \in \mathcal{V}$ represents an amino acid. To build edge connections, we first define a symmetric adjacency matrix $\boldsymbol{A}$ with the kNN-graph, i.e., each node is connected to up to k other nodes in the graph that has the smallest Euclidean distance over other nodes, and the distance is smaller than a certain cutoff, i.e., $30\mathring{A}$. Consequently, if v_i and v_j are connected to each other, we have $e_{ij} \in \mathcal{E}$ and $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji} \neq 0$.

A.2 NODE FEATURES

The node attributes $\boldsymbol{X} \in \mathbb{R}^{34}$ consist of chemical properties and geometric properties of amino acids. The chemical properties include:

- **residue type**. The wild-type proteins constitute 20 types of amino acid residues. We hereby take one-hot encoding on them and get the first $\boldsymbol{X}_{\text{aa}} \in \mathbb{R}^{20}$ node attributes.
- **standardized B-factor**. Crystallographic B-factor of the sum of the mainchain atoms describes the attenuation of X-ray or neutron scattering caused by thermal motion. The B-factor of α-carbon is a scalar value that is usually tested in laboratories to identify the rigidity, flexibility, and internal motion of each residue. Since the value is sensitive to the experimental environment and proteins in our dataset are measured by different laboratories, we take standardized B-factors along each protein to fix the measurement bias. For a given protein, we find the mean and standard deviation of the b-factors on each amino acid residue and normalize the raw b-factor values by deducting the mean value and then dividing the standard deviation. Consequently, 95% b-factor values are within the range between -1.8279 and 1.8081.

The geometric properties include:

- **SASA**. The solvent-accessible surface area measures the level of exposure of an amino acid to solvent in a protein (Heffernan et al., 2015). Since active sites of proteins are often located on their surfaces, SASA is regarded as an crucial structural property. We calculate SASA by the 'rolling ball' algorithm from its 3D structure. This algorithm uses a sphere (of solvent) of a particular radius to 'probe' the surface of the molecule. 95% SASA values are within the range between -1.9902 and 1.9614.
- **AA Position**. We use the position of α-carbon in each residue to record their 3D position $\boldsymbol{X}_{\text{pos}} \in \mathbb{R}^3$.
- **surface-aware node features**: we follow Ganea et al. (2021) and define 5 surface-aware node features by

$$\rho_i\left(\mathbf{x}_i; \lambda\right) = \frac{\left\|\sum_{i' \in \mathcal{N}_i} w_{i,i',\lambda}\left(\mathbf{x}_i - \mathbf{x}_{i'}\right)\right\|}{\sum_{i' \in \mathcal{N}_i} w_{i,i',\lambda}\left\|\mathbf{x}_i - \mathbf{x}_{i'}\right\|}, \text{ where } w_{i,i',\lambda} = \frac{\exp\left(-\left\|\mathbf{x}_i - \mathbf{x}_{i'}\right\|^2 / \lambda\right)}{\sum_{j \in \mathcal{N}_i} \exp\left(-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2 / \lambda\right)}$$

  we generate 5 surface-aware features by setting $\lambda \in \{1, 2, 5, 10, 30\}$.
- **dihedral angles**. To present the relative position of the amino acid in the protein chain, we calculate the trigonometric values of dihedral angles (Li et al., 2022) $\{\sin, \cos\} \circ \{\phi, \psi\}$ from the backbone atom positions. For a specific v_i of the ith amino acid in the protein sequence, the dihedral angles are measured from 3D positions of $C\alpha_{i-1}, N_i, C\alpha_i, N_{i+1}$. The resulting feature $\boldsymbol{X}_{\text{agl}} \in \mathbb{R}^4$. Note that we remove the end node due to the missing angles.

A.3 EDGE ATTRIBUTES

The edge attributes $\boldsymbol{E} \in \mathbb{R}^{93}$ are featured on the connected edges, including

- **positional encoding** For the two nodes v_i and v_j, we encode their sequential relationship by their relative position $d_{i,j} = |s_i - s_j|$, where s_i and s_j are the absolute positions of the two nodes in the amino acid chain. For instance, if v_i is the first amino acid in the sequence and v_j is the fifth amino acid, we have $s_i = 1$ and $s_j = 5$. As the locally connected nodes (by the kNN defined edges) merely have their positional distance further than 64, we set a cutoff at 64, i.e., $d_{i,j} = \min(|s_i - s_j|, 65)$. We follow Liu et al. (2022) and transform this distance feature with one-hot encoding. In addition, we define a contact signal (Li et al., 2022) to indicate whether two residues contact in the space, i.e., the Euclidean distance $\|C\alpha_i - C\alpha_j\| < 8$. In total, we create 66 binary features for the relative position of two connected nodes (by graph edges).
- **inter-residue distances**. We use Gaussian radial basis functions (RBF) of inter-residue distances as additional edge attributes. For the edge between node i and node j, the distance reads

$$\boldsymbol{E}_{\text{rbf}} = \exp\left\{\frac{(\|\boldsymbol{x}_j - \boldsymbol{x}_i\|)^2}{2\sigma_r^2}\right\},\ r = 1, 2, \ldots, R$$

  In aligned with Ganea et al. (2021), we set the scale parameter $\sigma_r = \{1.5^k | k = 0, 1, 2, \ldots, 14\}$. In total, there are 15 distinct distance-based features on each edge.
- **local frame orientation** is calculated from heavy atoms positions in these two residues. It represents local fine-grained relations between amino acids and the rigid property of how these two residues interact with each other.

B Deep Mutational Scanning Benchmark

In order to validate the performance of our pre-trained method on mutant effect prediction, we test the pre-trained methods on a diverse set of proteins from Deep mutational scanning (DMS) experiments, which provide a systematic survey of the mutational landscape of proteins from wet laboratory test and is usually used to benchmark computational predictors for the effects of mutations.

This section introduces the main aspects that help better understand the task, including the test dataset description, the pre-processing steps, as well as the evaluation details.

B.1 Dataset Glossary

On the mutant effect prediction task, we evaluate model performance on $199,819$ records from 15 *in vivo* and *in vitro* DMS experiments that cover 1-site to 28-sites mutant scores, where 6 of them only mutant on single sites, and 15 of them have both single site and higher-order sites DMS. Specifically, we collect all the single-site DMS (**BG_STRSQ**, **BLAT_ECOLX**, **CALM1_HUMAN**, **HG_FLU**, **KKA2_KLEPN**, **MTH3_HAEAESTABILIZED**, **PA_FLU**, **PTEN_HUMAN**, and **RASH_HUMAN**) and two proteins with deep mutants (**F7YBW8** and **F7YBW8_MESOW**) from Riesselman et al. (2018)'s work; **GFP** by Sarkisyan et al. (2016); **CAPSD_AAV2S**, **DLG4_HUMAN**, and **GRB2_HUMAN** in Notin et al. (2022)'s research.

Essentially, the dataset provides these protein sequences, mutant actions, and fitness scores on different mutants. Due to the lack of experimentally tested structures, we use AlphaFold (Jumper et al., 2021) to predict their structures. Since we only focus on AA type change mutant actions, there are only 0.265% (470 out of $200,349$) mutant actions changing the length of proteins, so we removed them. The fitness scores reflect measurable features of the protein with respect to certain mutations, such as enzyme function, growth rate, peptide binding, viral replication, and protein stability. A higher fitness score implies that the mutant protein is better off after the adjustment of some sidechain types. The graph construction method and feature attraction process are exactly the same as we did on training dataset, except that for the convenience of later correlation computation, we append the mutant actions and fitness scores as its graph features. Table 3 summarizes the characterization of each mutational scanning dataset, including the protein length and the number of scores they recorded in different orders of mutations.

We also investigate the choice of mutant order to the test score in DMS datasets. As we are more interested in the rank of mutants than their absolute scores, we rank the score values in each of the proteins and make boxplots on them. In other words, to renovate a given protein to perform better

Table 3: Summary of the Higher-Order Mutant Test Dataset.

# mutant(s)	CAPSD_AAV2S	GFP	F7YBW8	F7YBW8_MESOW	DLG4_HUMAN	GRB2_HUMAN
# node	734	235	92	92	723	216
1	1,064	1,084	37	37	1,280	1,034
2	21,666	12,777	499	499	5,696	62,332
3	13,812	12336	2798	2798		
4	13,292	9,387	5,858	5,858		
5	12,596	6,825				
6	10,792	4,298				
7	1,716	2,526				
8	1,478	1,364				
9	1,302	627				
10	1,166	299				
11	890	118				
12	814	43				
13	736	23				
14	656	5				
15	572	2				
16	472					
17	406					
18	318					
19	238					
20	186					
21	148					
22	112					
23	86					
24	58					
25	34					
26	24					
27	16					
28	6					
sum	84656	51714	9192	9192	6976	63366

on a certain property, the directed evolution with a higher score (or equivalently a higher rank or smaller rank score) is preferred. As shown in Figure 1, the proteins that were validated in this work generally present a negative relationship between the mutant order and rank score. That is, a higher mutant order results in a smaller rank score, i.e., a higher rank.

B.2 Test Task

We evaluate performance by comparing the experimental ground truth fitness score with the predicted score for each deep mutational scan using Spearman's rank correlation.

For a specific mutation of interest, we score it by the log odds ratio from the probabilities of the sidechain type classification task with respect to wild-type probabilities (Meier et al., 2021; Lu et al., 2022). When the higher-order (double-site or more sites) mutations exist in a single protein sequence, we assume an additive model over the mutated positions. To be specific, for T-site mutants, the fitness score reads

$$\sum_{t \in T} \log p(\boldsymbol{x}_{\text{aa}} = \hat{\boldsymbol{x}}_{\text{aa}}^{\text{mutant}}) - \log p(\boldsymbol{x}_{\text{aa}} = \boldsymbol{x}_{\text{aa}}^{\text{wild}}), \tag{5}$$

where $\hat{\boldsymbol{x}}_{\text{aa}}^{\text{mutant}}$ and $\boldsymbol{x}_{\text{aa}}^{\text{wild}}$ denote the predicted sidechain type, and the wild-type sidechain type, respectively.

In the main experiments, we validate the prediction performance with different problem setups. Depending on the different degrees of freedom on the mutation, we consider respectively arbitrary order of mutants (single-site or multiple-sites), higher-order mutants (multiple-sites only), or fixed-order mutants (n-sites with a particular order n). For instance, when investigating the prediction performance of higher-order mutants on a given protein, we first make predictions on all the DMS that has two or more mutant sites. The Spearman correlation coefficient is then calculated with the predicted and experimental score sequences. Alternatively, if the number of mutations is specified to 3-sites, only DMS containing 3 mutations will be included for scoring.

Table 4: Summary of the generated graph datasets by **CATH** for training dataset and **Mutant** for the test dataset. We consider three variants of graphs by **CATH** with three different ks (i.e., $k = 5, 10, 20$) for generating the kNN-graphs. For the test dataset, we report the statistics for the 8 proteins with higher-order mutants.

		# feature		**# node**				**# edge**		
dataset	# graph	node	edge	min.	max.	avg.	avg. D	min.	max.	avg.
CATH-s40-k10	31,848	34	93	8	1,201	150.92	9.96	56	12,004	1,503.62
CATH-s40-k5	31,848	34	93	8	1,201	150.92	4.98	38	6,004	751.99
CATH-s40-k20	31,848	34	93	8	1,201	150.92	19.92	56	24,009	3,006.31
Mutant-Nsite-k10	15	34	93	92	734	365.73	9.99	916	7,333	3,652.27
Mutant-Nsite-k5	15	34	93	92	734	365.73	4.99	457	3,668	1,826.27
Mutant-Nsite-k20	15	34	93	92	734	365.73	19.94	1,830	14,656	7,293.93

C Training Details

C.1 Dataset for Pre-training

CATH (Orengo et al., 1997) prepares a diverse set of proteins with experimentally determined 3D structures from the Protein Data Bank (PDB) and, where applicable, splits them into their consecutive polypeptide chains. We employ a non-redundant subset of **CATH v4.3.0** domains for pre-training the model. No pairs of domains in the selected protein entities have more than 40% sequence identity over 60% of the overlap (over the longer sequence in the protein pair of comparison).

The revised **CATH** datasets contains over 30,000 sample protein sequence. We then transform each of them into a protein graph, as is defined in Appendix A. We summarize the main properties of the **CATH** dataset in the first three lines of Table 4, where we use **s40** to denote the sequence identity, and **k5, k10** and **k20** to represent the number of neighbors in generating the kNN-graphs. Similar progress of graph generation is adopted to the test protein sequences (including 9 single-site DMS proteins and 6 multiple-sites DMS proteins), which statistics are attached in the table as well.

C.2 Model Setup

This section discloses the full experimental details, including data preparation, access to model implementation, and their tuning space. All the experiments are conducted with PyTorch on NVIDIA ® RTX 3090 GPU with 10,496 CUDA cores and 24GB memory on an HPC cluster. The models are programmed on PyTorch-Geometric (version 2.0.1) and PyTorch (version 1.7.0).

Program We take the official implementation of the baseline models from the repository:

- MSA Transformer: `https://github.com/facebookresearch/esm`
- ESM-1v: `https://github.com/facebookresearch/esm`
- ESM-IF1: `https://github.com/facebookresearch/esm`
- DeepSequence: `https://github.com/debbiemarkslab/DeepSequence`
- Tranception: `https://github.com/OATML-Markslab/Tranception`
- ProGen2: `https://github.com/salesforce/progen`

The EGC layers are implemented with the official PyTorch implementation at `https://github.com/lucidrains/egnn-pytorch`. All the pre-trained baseline models are released in the GitHub repository. In particular, ESM-1v has 5 variants with different setups and learned parameters, for which we run the test on all the versions and take average performance on them. Our program will be published upon acceptance.

Hyper-parameters Setting The model architecture stacks with 6 EGC layers and a linear classifier to make predictions. We use Adam optimizer to optimize our model without a warmup period. We train our model for 300 epochs with the initial learning rate 0.001 and weight decay 0.01. After

150 epochs, the learning rate is decay to 0.0001. We use gradient clipping equal to 4 in order to stabilize the training procedure.

D PRIOR BIOLOGICAL KNOWLEDGE

This section investigates the influence of adopting prior biological knowledge, including the choice of noise distribution in perturbing the sidechain type, and the implementation of label smoothing. We analyze the effect of these designs with the experimental results.

For the distribution of sidechain type corruptions, we consider three particular types of noise, including random perturbation, wild-type-based perturbation, and BLOSUM-based perturbation. The results are reported in Table 5. Here we fix the confidence level $p = 0.6$ and the penalty weight $\lambda = 0.2$ for all the loss items, except for dihedral loss, which λ is set to 0.5. The influence of selecting different ps will be discussed in the next section.

Table 5: Test Performance with the three different sidechain perturbation types (random, wild-type-based, and BLOSUM substitution matrix-based) on the fitness of mutant effect prediction. The average Spearman's rank coefficients are reported on both single-site and multiple-sites mutations.

	RANDOM		WILD-TYPE	
Learning Task	single	multi	single	multi
AA	0.1551	0.3350	0.1482	0.3357
AA+SASA	0.2734	0.4909	0.2747	0.5037
AA+SASA+B-factor	0.2705	0.4480	0.2803	0.4593
AA+SASA+coordinates	0.2445	0.4245	0.2364	0.4290
AA+SASA+B-factor+coordinates+dihedral	0.1752	0.3657	0.1709	0.3345

E EFFECT OF THE CONFIDENCE LEVEL

This section discusses the choice of the confidence level p in Equation 1, i.e., a tunable parameter that controls the proportion of residues that are 'noise-free'. We begin with demonstrating the noise level of the sidechain type by visualizing the perturbed amino acid residues amount. While this value can be determined by humans which reflects their belief in the quality of wild-type proteins, this research focuses on data-driven decisions, i.e., we conduct experiments on different levels of ps to guide an empirical choice of it.

E.1 NOISY RATIO ON THE AA TYPE

Figure 7-8 demonstrates different perturbation levels on the AA type with wild-type noises and BLOSUM matrix.

In Figure 7, each of the bar charts visualizes the probability distribution of the perturbed amino acid residues in an epoch. For instance, $p = 1$ indicates the maximum level of confidence in the quality of wild-type proteins, resulting in no perturbations in the input amino acid residues. In contrast, $p = 0.1$ gives a total number of $90,265$ corruptions in a training epoch, where 998 of them become Cysteine (abbreviated as C), and $9,194$ of them are corrupted to Leucine (abbreviated as L).

Figure 8 demonstrates the modified **BLOSUM** matrix with different temperatures for defining the label smoothing and perturbation probability. A higher temperature is agnostic to a higher confidence level p, resulting in a more diagonal substitution matrix. We report the deep-mutant prediction performance over the 6 multi-site mutant proteins with $p = 0.6$. The output task predicts AA type, SASA, and B-factor with all the λs fixed to 0.2. We report the average Spearman's correlation over 5 repetitive runs by applying the BLOSUM matrix to label smoothing and the distribution of noisy input AA types. The results are reported in the titles of respective subfigures.

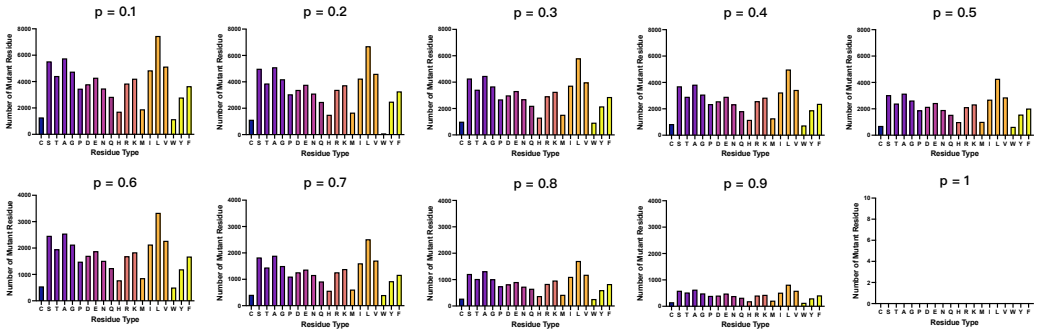

Figure 7: Perturbed label distribution of the training protein graphs at different confidence levels p. A higher confidence level results in fewer AAs to observe noisy labels.

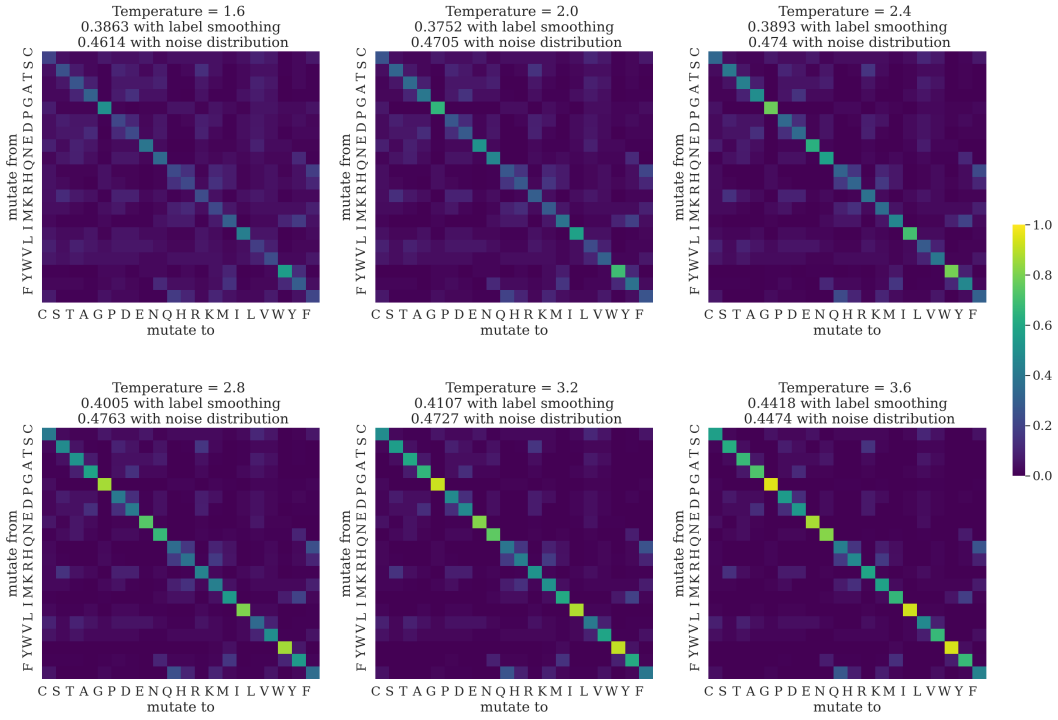

Figure 8: Perturbed label distribution of the training protein graphs at different temperatures. As the BLOSUM matrix has been applied to perturbation distribution and label smoothing of AA types, we report the average Spearman's correlation on the titles of sub-figures over 5 repetitive runs.

Table 6: Average Spearman's rank coefficients at different confidence levels p with different types of probability distribution on **single-site** mutant effect prediction. We fix the number of EGC layers to 6 and $\lambda = 0.2$ for all the losses except for dihedral, which λ we set to 0.5.

	Learning Task	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
RANDOM	AA	0.1391	0.1809	0.1816	0.1551	0.1445	0.1557	0.1517	0.1219
	AA+SASA	0.2479	0.2603	0.2869	0.2734	0.2933	0.2840	0.2902	0.1354
	AA+SASA+B-factor	0.2745	0.2679	0.2790	0.2705	0.2767	0.2642	0.2755	0.1321
	AA+SASA+coordinates	0.2458	0.2493	0.2239	0.2445	0.2251	0.2507	0.2045	0.1778
	AA+SASA+B-factor+coordinates+dihedral	0.1820	0.1528	0.1594	0.1752	0.1620	0.1798	0.1515	0.1312
WILD-TYPE	AA	0.1715	0.1600	0.1597	0.1842	0.1745	0.1674	0.1679	0.1406
	AA+SASA	0.2663	0.2660	0.2662	0.2747	0.2797	0.2894	0.2897	0.1382
	AA+SASA+B-factor	0.2708	0.2717	0.2519	0.2803	0.2792	0.2673	0.2776	0.1351
	AA+SASA+coordinates	0.2539	0.2107	0.2595	0.2364	0.2474	0.2230	0.2628	0.1818
	AA+SASA+B-factor+coordinates+dihedral	0.1771	0.1709	0.1917	0.1709	0.1937	0.1495	0.1769	0.1566

E.2 INFLUENCE OF p ON THE PRE-TRAINED MODEL

As mentioned before, the choice of p can be determined by prior knowledge regarding the quality of wild-type proteins. Alternatively, this value can be considered as a data-driven hyper-parameter to be optimized during the model training. We hereby follow the second path and search for the optimal $p \in \{0.3, 0.4, \ldots, 0.9, 1\}$. Here we exclude extremely small ps to avoid drastic perturbation rates. The influence on the different choices on ps are validated with different learning tasks (i.e., we consider different outputs) and with different types of perturbation distribution (i.e., random perturbation, wild-type perturbation, and BLOSUM matrix-based perturbation). We report the results of average Spearman's ρ on different variants in Table 6 (single-site mutants) and Table 7 (higher-order mutants). In general, a moderate p between 0.3 and 0.6 best suits the majority selection of learning modules and noise distribution. Based on the overall performance, we suggest $p = 0.6$ as the default value of the confidence level.

F DESIGN OF THE MULTI-TASK LEARNING PROBLEM

This section discusses different choices on the prediction tasks. Recall in Section 2 we introduce five learning tasks on sidechain type prediction, SASA and b-factor regression, 3d-coordinate recovery, and dihedral angle prediction. Here we aim at answering two questions:

1. which learning targets are preferred over others?
2. which λ(s) should be set as the default value(s)?

F.1 CHOICES ON THE PREDICTIONS

To answer the first question, we revisit the results in the previous sections. To enable a perceptual presentation, we visualize their performance in Figure 5 and Table 7-8 in the main table with different ps and λs and compare them with baseline methods. As a default choice, predicting AA type, SASA and B-factor helps generate representative protein graph embedding for further tasks.

F.2 CHOICES OF LOSS WEIGHT

We next detail the influence of the loss function to the overall model performance by selecting different λs. For simplicity, we let $\lambda_1 = \lambda_2 = \lambda_3 \in \{0.05, 0.1, 0.2, 0.5, 0.8, 5, 10\}$. Similar to before, we fix $\lambda_4 = 0.5$ and $p = 0.6$. The probability distribution on the sidechain type uses the wild-type distribution. The weighted average performance on single-site and multiple-site mutant effect predictions are reported in Table 8.

Table 7: Average Spearman's rank coefficients at different confidence levels p with different types of probability distribution on **higher-order** mutant effect prediction. We fix the number of EGC layers to 6 and $\lambda = 0.2$ for all the losses except for dihedral, which λ we set to 0.5.

	Learning Task	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
RANDOM	AA	0.3145	0.3435	0.3709	0.3350	0.3266	0.3264	0.3135	0.2521
	AA+SASA	0.5023	0.4285	0.4986	0.4909	0.4749	0.4686	0.4434	0.2363
	AA+SASA+B-factor	0.4707	0.4724	0.4741	0.4480	0.4372	0.4139	0.4369	0.2810
	AA+SASA+coordinates	0.4355	0.4763	0.3962	0.4245	0.4405	0.4078	0.3842	0.2809
	AA+SASA+B-factor+coordinates+dihedral	0.3554	0.3293	0.3087	0.3657	0.2871	0.3070	0.3702	0.2420
WILD-TYPE	AA	0.3184	0.3189	0.3357	0.3357	0.3557	0.3343	0.3189	0.2482
	AA+SASA	0.4717	0.4548	0.4777	0.5037	0.4578	0.4763	0.4740	0.2716
	AA+SASA+B-factor	0.4627	0.4704	0.4426	0.4593	0.4678	0.4527	0.4498	0.2512
	AA+SASA+coordinates	0.4351	0.3839	0.4648	0.4290	0.4536	0.4327	0.4311	0.2851
	AA+SASA+B-factor+coordinates+dihedral	0.4151	0.3932	0.3596	0.3345	0.3432	0.2974	0.2937	0.2673

Table 8: Test Performance of (model) with different λs on the mutant effect prediction.

	Learning Task	0.05	0.1	0.2	0.5	0.8	5	10
single	AA	0.1719	0.1752	0.1621	0.1644	0.1648	0.1714	0.1554
	AA+SASA	0.2953	0.2910	0.3023	0.2834	0.3002	0.2782	0.2709
	AA+SASA+B-factor	0.2975	0.3084	0.3032	0.2884	0.2744	0.2622	0.2329
	AA+SASA+coordinates	0.2654	0.2788	0.2604	0.2511	0.2785	0.2260	0.2079
	AA+SASA+B-factor+coordinates+dihedral	0.1844	0.1880	0.1366	0.1825	0.1641	0.1556	0.1933
multiple	AA	0.3095	0.3430	0.3441	0.3324	0.3326	0.3326	0.3147
	AA+SASA	0.4738	0.4731	0.4678	0.4541	0.4010	0.4500	0.4022
	AA+SASA+B-factor	0.4648	0.5149	0.4501	0.4270	0.4811	0.4421	0.4149
	AA+SASA+coordinates	0.4748	0.4782	0.4398	0.4550	0.4545	0.3980	0.3935
	AA+SASA+B-factor+coordinates+dihedral	0.3326	0.3820	0.3785	0.3465	0.3361	0.3260	0.3396