# Lightweight Equivariant Graph Representation Learning for Protein Engineering

**Bingxin Zhou**
The University of Sydney
bzho3923@uni.sydney.edu.au

**Outongyi Lv**
Shanghai Jiao Tong University
harry_lv@sjtu.edu.cn

**Kai Yi**
University of New South Wales
kai.yi@unsw.edu.au

**Xinye Xiong**
Shanghai Jiao Tong University
cintia_bear@sjtu.edu.cn

**Pan Tan**
Shanghai Jiao Tong University
tpan1039@alumni.sjtu.edu.cn

**Liang Hong**
Shanghai Jiao Tong University
hongl3liang@sjtu.edu.cn

**Yu Guang Wang**
Shanghai Jiao Tong University
yuguang.wang@sjtu.edu.cn

## Abstract

This work tackles the issue of directed evolution in computational protein design that makes an accurate prediction for the function of a protein mutant. We design a lightweight pre-training 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 task for the higher-order mutants, where many amino acid sites of the protein are mutated, the proposed training 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 [30, 39, 49] or enzymes [37, 48]. *Directed evolution* aims at optimizing a protein's functional fitness, where a greedy search is usually conducted in the local sequence along the hundreds to thousands of AA sites in a protein to mutate to proper AA types over 20 candidates to render a protein mutant with the highest gain-of-function [34]. Normally, multiple AA sites ($\sim$5-10) of the protein need to be mutated to obtain a mutant with great fitness [1, 8], namely *deep mutants*. However, the astronomical number of potential combinations in deep mutants prevent systematic experiments from testing on all possible deep mutants.

Alternatively, *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

Machine Learning for Structural Biology Workshop, NeurIPS 2022.

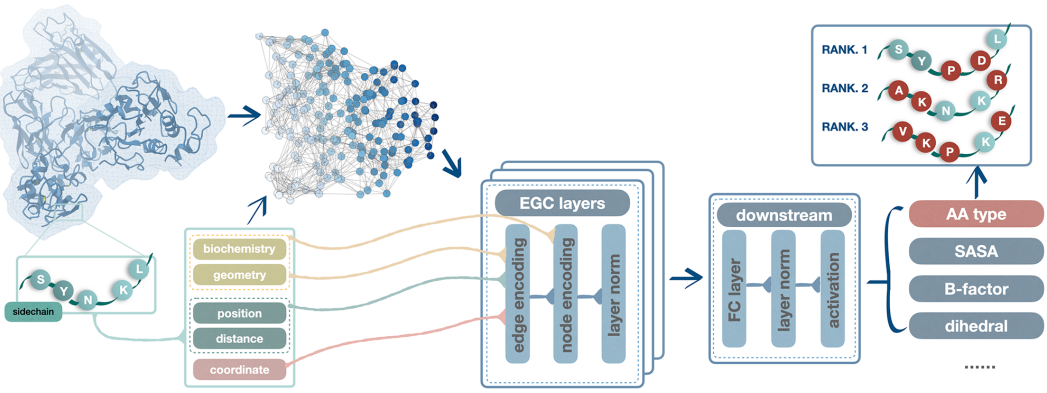


Figure 1: LGN is pre-trained with protein graphs of (perturbed) node and edge attributes, as well as the nodes' 3D positions with a multi-task learning strategy. The structural inputs are encoded with EGC layers to extract translation invariant and rotation equivariant representations for each node on individual graphs. The downstream tasks employs fully-connected layers to learn different labels, where the log-odd-ratios of AA type predictions are used for suggesting top-ranked mutations.

[23, 37, 44]. Due to the scarcity of labeled protein data, researchers often pre-train on protein sequences or structures to learn protein encoding for downstream tasks, such as *de novo* protein design [16] and higher-level structure prediction [9]. In the context of fitness prediction of mutation effect, existing methods usually transform the problem to mini-*de novo* design, i.e., inferring a specific AA type from its micro-environment, including the neighborhood AA types. Current state-of-the-art protein sequence-based methods rely heavily on multiple sequence alignment (MSA) [11, 31, 32] and protein language models [4, 9, 26, 33]. While MSA helps capture important evolutionary properties of the protein family, it 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 billions of protein sequences. Meanwhile, an auto-regressive inference process is usually needed along the entire protein sequence to score a mutation on a single site, which further increases the inference cost [16, 21, 25, 27, 37]. More importantly, when predicting the fitness of deep mutants, a majority models made a crude assumption that the multiple-site mutation effect is a linear summation of the effect of each individual mutation, which is incorrect in most cases [5, 19]. The ignored epistatic effects between different sites is potentially a key factor hindering the acquisition of favorable high-order mutants in directed evolution [35, 36].

Mutation of AA sites also occurs in nature, and it is suggested by natural selection that only the mutation that exhibits the best fitness and fits the environment survives. Altering AA types of a protein in nature can be viewed as adding corruptions to the node features of the protein graph, and denoising the graph makes a remedy to search for deep mutants with the best fitness. We hereby model the protein mutation effect prediction as a denoising problem with equivariant graph neural networks [38]. We generate *protein graph*s as an elegant structural description of the raw protein that encodes the graph geometry. Each protein is treated as a graph with AAs being graph nodes with the first-level information, such as AA types and spatial coordinates of C$\alpha$, being node features. The learning scheme predicts the fitness of mutation effect on a protein with an arbitrary number of mutant sites and provides efficient recommendations to discover favorable mutants. Compared to existing zero-shot state-of-the-art methods for mutation effect predictions, the designed lightweight equivariant graph neural network (LGN) stands out in three perspectives.

1. Implementing the multi-task learning strategy and biological priors enhance LGN's generalization ability. The former trains data-driven black-box encoders to describe a protein's microenvironment, and the latter inserts domain knowledge towards practically meaningful representations.

2. LGN is efficient in training and inference. The spatial graph inputs portray the topological properties of proteins that circumvent data augmentation.Equivariant message passing layers of the model, alternatively, provides a feature distillation unit with translation invariance and rotation equivariance [2], which can precisely capture the micro-frame property in the protein graph geometry. Meanwhile, the downstream task allows outputting the probability of all the AAs at a time that avoids iterative inference in auto-regression.

3. The typical approach to calculating the higher-order mutation effect sums up log-odd-ratio scores of the corresponding individual single-site mutants [25, 16]. However, the linear summation is unsubstantiated, as the epistatic effect is known to be present that mutations of individual sites are not independent [19, 5]. Instead, our scoring matrix does not need any of the independent-mutation assumptions, and the output can take any number of mutation sites at a time.

## 2 Zero-shot Learning for Protein Recovery

### 2.1 Graph Representation of Protein Structure

We create a k-nearest neighbor (kNN) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to describe a given protein's 3D structure and molecular properties (See Appendix B). A node $v_i \in \mathcal{V}$ represents an AA residue with node attributes constituting biochemical and geometric properties. The former includes 20 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 $\boldsymbol{X}_{\text{pos}} \in \mathbb{R}^3$ from 3D coordinates of the $\alpha$-carbon in a residue and the relative position of the amino acid in the protein chain ($\boldsymbol{X}_{\text{agl}} \in \mathbb{R}^4$) by the dihedral angles computed from the backbone atom positions. The edge attributes $\boldsymbol{E} \in \mathbb{R}^{93}$ are featured on the connected edges, including 15 inter-atomic distances, 12 local N-C positions, and 66-dimensional position encoding that records the relative position in the protein sequence.

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

Wild-type proteins suffer from random perturbations or mutations that not every AA site has the best fitness. We pre-train LGN with a multitask learning strategy that removes the natural corruptions and predicts key protein properties to help encode the microenvironment of recovered proteins.

**AA Type Denoising** We refine a node of AA type $\boldsymbol{x}_{\text{aa}}$ to $\tilde{\boldsymbol{x}}_{\text{aa}}$ by adding 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 investigates three types of noise distribution, including random distribution, wild-type protein distribution[1], and BLOSUM-matrix based perturbation, which we shall introduce shortly.

**Geometric Properties Denoising** For continuous-valued features, such as 3D coordinates and dihedral angles, an i.i.d Gaussian noise is introduced, e.g., $\tilde{\boldsymbol{x}}_{\text{pos}} = \boldsymbol{x}_{\text{pos}} + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, I_3)$. Denoising the spatial features approximates the data-generating force field of molecules [50].

**Bio-chemistry Properties Recovery** Aside from the perturbed residues type and geometric properties to denoise, other auxiliary tasks are introduced to better retrain the 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 [21]. We thus introduces inductive biases to the model by predicting these two properties in the downstream tasks.

**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 the BLOSUM substitution matrix [15] to account for the relative substitution frequencies and chemical similarity of amino acids, which is derived from the statistics for every conserved regions 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 modify our loss function that a mutation to an AA type with a higher similarity score accumulates a smaller penalty than to the one with lower similarity score.

[1]The probability distribution is retrieved from AlphaFold [47] at `https://alphafold.ebi.ac.uk/`

### 2.3 Protein Structure Representation with Equivariant GNNs

As the initial proteins are structured in the three-dimensional space, it is vital for the model to predict the same binding complex no matter how the input proteins are positioned and oriented. Rather than practicing expensive data augmentation strategies, we construct SE(3)-equivariant and invariant neural layers for graph embedding. At the $l$th layer, an Equivariant Graph Convolution (EGC) [38] inputs an $n$ hidden node properties embedding of a graph $\boldsymbol{H}^l = \left\{\boldsymbol{h}_1^l, \ldots, \boldsymbol{h}_n^l\right\}$ and the node coordinate embeddings $\boldsymbol{X}_{\text{pos}}^l = \left\{\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_n^l\right\}$. The attributed edges are denoted as $\boldsymbol{E} = \{\ldots, \boldsymbol{e}_{ij}, \ldots\}$. Concisely: $\boldsymbol{H}_{\text{pos}}^{l+1}, \boldsymbol{X}^{l+1} = \text{EGC}\left[\boldsymbol{H}^l, \boldsymbol{X}_{\text{pos}}^l, \boldsymbol{E}\right]$, where the EGC layer reads

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

The edge and node representations are propagated by $\phi_e, \phi_h$, such as multi-layer perceptrons (MLPs). The additional operation 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 equivariance to permutations on $\mathcal{V}$.

### 2.4 Model Overview

Our model is depicted in Figure 1. The downstream tasks include AA type classification, SASA and B-factor prediction, 3D coordinates denoising, and dihedral angle denoising. The total loss reads

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{aa}} + \lambda_1\mathcal{L}_{\text{sasa}} + \lambda_2\mathcal{L}_{\text{b-fac}} + \lambda_3\mathcal{L}_{\text{pos}} + \lambda_4\mathcal{L}_{\text{agl}}, \tag{3}$$

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). The classification loss $\mathcal{L}_{\text{aa}}$ for AA type is measured by cross-entropy with *label smoothing* [43]. On an arbitrary node $i$,

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

where $p(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ denotes the true distribution and $q_\theta(\hat{y}_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ is the predicted label distribution following a softmax function. To improve the generalization and respect the prior biological knowledge, we modify the true 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 $\hat{y}_{\text{aa}} = y_{\text{aa}}$ and $\varepsilon u(y_{\text{aa}}|\boldsymbol{X}_i, \boldsymbol{E}_i)$ otherwise with some tolerance factor $\varepsilon$. The distribution of $u(y|x_i)$ is defined by the BLOSUM substitution matrix.

## 3 Fitness of Mutation Effect Prediction

### 3.1 Experimental Setup

Our LGN is pre-trained on **CATH v4.3.0** [29] with artificial noise to predict AA type, 3D coordinates, dihedral angles, and chemical properties (SASA and B-factor). We will examine the model performance with different output tasks. For instance, LGN (AA+SASA) indicates we predict AA types and SASA in the output. The evaluation is conducted with deep mutational scanning (DMS) datasets for zero-shot fitness of mutation prediction tasks. The model performance is compared against popular state-of-the-art language models and structure-enhanced models, including MSA TRANSFORMER [31], ESM-1V [25], and ESM-IF1 [16].

**Lightweight Equivariant Graph Neural Networks (LGN)** We generate protein graphs for the sequences in **CATH** for pre-training LGN. We assign random perturbations to AA types and other features in Section 2. At the validation step, the noises are fixed for stable and comparable measurements. The main architecture constitutes 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 AAs, 1 predicted SASA, 1 B-factors, and 4 dihedral values (when applicable). The 3D-coordinates are derived directly from EGC outputs.

**Evaluation** All the models are evaluated with deep mutational scanning (DMS) assays on 15 proteins, including 9 proteins considering single-site mutant, and 6 proteins with deep mutants (see Appendix C). We do not append artificial noises onto the test proteins, as we assume they are already noisy. The unmutated test proteins are sent to the pre-trained LGN model and use output the log-odd-ratio in Equation 4 of the predicted probabilities of AA types for suggesting the deep mutant ranks. The final performance on deep mutant prediction is evaluated on Spearman's correlation coefficient between the predicted and experimental scores on all the mutation combinations.

### 3.2 Results Analysis

**Fitness of Deep Mutants Prediction** We first evaluate the fitness of proteins' mutation effects prediction with zero-shot models, i.e., the fitness scores are inferred directly from a pre-trained model without fine-tuning on a task-specific model. We visualize the overall performance comparison on protein-wise Spearman correlation coefficients in Figure 2. LGN outperforms baseline methods in deep mutant tasks and achieves at least comparable results in single-site mutant tests with an overall 0.5037 weighted average correlation by predicting AA types and SASA. In comparison, the scores for MSA TRANSFORMER, ESM-1V, and ESM-IF1 are 0.3756, 0.3902, and 0.4270, respectively.

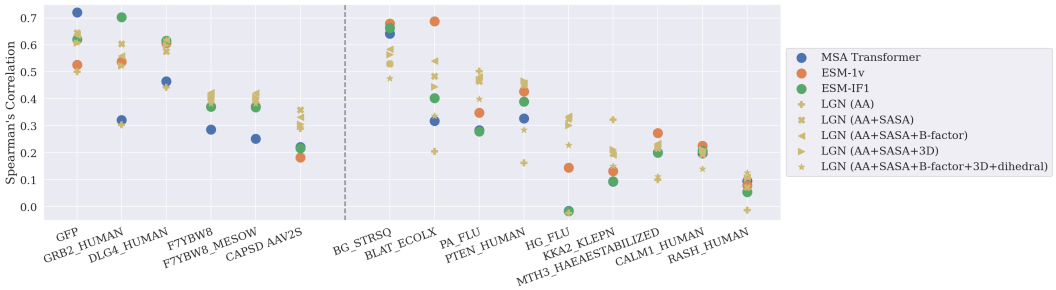


Figure 2: Per task Spearman's correlation coefficients on the fitness of deep mutant effect prediction on different protein datasets (denoted on the x-axis) with zero-shot models. The left 6 proteins contain deep mutations, and the right 9 proteins only record shallow DMS.

**Protein Recovery** We next investigate the model's fitness in the auxiliary learning tasks in predicting SASA, and B-factor. We examine the $R^2$ of the predicted SASA and B-factor and the ground-truth values on **CATH** and report the results in the middle and right subplots in Figure 3. 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 0.000. In addition, Pearson's correlation coefficients for the two variants are respectively 0.884 and 0.791.

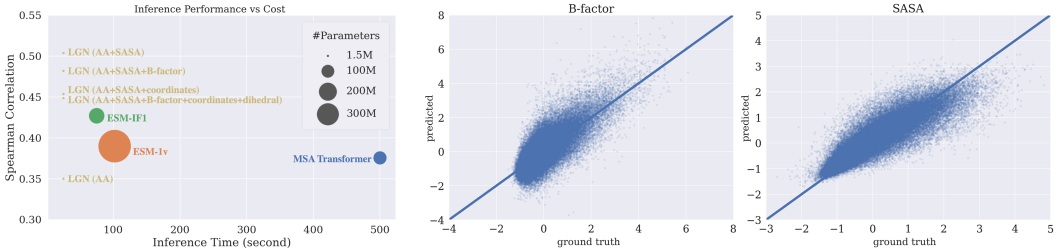


Figure 3: **Left**: Comparison of Inference Efficiency. **Middle & Right**: Regression performance of the pre-trained model on the perturbed proteins.

**Inference Speed** LGN consumes significantly fewer computational resources in training and inference. We make a direct comparison of the model scale, inference time, and prediction performance and visualize the results in the left subplot of Figure 3. The radius of each ball indicates the number of network parameters of a model. Our model with different output variants (in yellow) can achieve SOTA performance on Spearman's correlation (y-axis) with the minimum inference time (x-axis) while requiring less than 1% fitting parameters of ESM-IF1 and MSA TRANSFORMER's. The gap on ESM-1V is larger, which requires more than 400 times of parameters than ours.

## 4 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 costing, 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 C$\alpha$) from observed noisy proteins. We employ translation invariant and rotation equivariant neural message passing layers to extract the invariant geometry features of micro frames within and between 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


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

[2] Kenneth Atz, Francesca Grisoni, and Gisbert Schneider. Geometric deep learning on molecular representations. *Nature Machine Intelligence*, 3(12):1023–1032, 2021.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] Yinglu Cui, Yanchun Chen, Xinyue Liu, Saijun Dong, Yu'e Tian, Yuxin Qiao, Ruchira Mitra, Jing Han, Chunli Li, Xu Han, et al. Computational redesign of a petase for plastic biodegradation under ambient condition by the grape strategy. *ACS Catalysis*, 11(3):1340–1350, 2021.

[9] 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*, pages 1–1, 2021.

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

[11] 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.

[12] 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.

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

[14] 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.

[15] 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.

[16] 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, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8946–8970. PMLR, 17–23 Jul 2022.

[17] 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.

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

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

[20] 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 preprint arXiv:2201.13299*, 2022.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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, pages 29287–29303, 2021.

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

[27] 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, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16990–17017. PMLR, 17–23 Jul 2022.

[28] 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.

[29] 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.

[30] 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.

[31] 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*, pages 8844–8856. PMLR, 2021.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

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

[39] 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.

[40] 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.

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

[42] 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*, pages 20503–20521. PMLR, 2022.

[43] 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*, pages 2818–2826, 2016.

[44] 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.

[45] 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.

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

[47] 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.

[48] 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.

[49] 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.

[50] 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.

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

# A Additional Related Work for *in silico* Protein Engineering

**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 [10, 16]. 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 [7, 25, 28]. However, the geometry of proteins also suggests higher-level structures and topological relationships that are vital to protein function. Structure prediction of proteins always attracts great attention in the field [6, 17, 3, 47]. The breakthrough progress in protein folding also enriches structured proteins. For instance, [16] and [24] 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 changed no matter how a protein is translated or rotated from one place to another [13]. 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 [46, 37]. 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 [12, 42].

**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 [45]. Contrastive learning and self-prediction [9], [51], and[16] used self-supervised pre-training methods provide ways to extract a good representation for reducing computational resources. Despite only applying classical representation learning methods on protein, some researchers designed sophisticated methods. [20] proposed $\vec{W}$-GNN variants efficiently interact scalar-vector features. Besides, [41] introduced a multi-scale model HOLOPROT connecting surface to structure and sequence.

**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 [32], ALPHAFOLD [17] and MSA TRANSFORMER [31]. The MSA for a protein sequence or domain captures meaningful information on 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 [40], 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 [25] trains a zero-shot model on a large set of unaligned sequences to secure a scalable and bias-free training procedure, and TRANCEPTION [27] leverages autoregressive predictions and retrieval of homologous sequences at inference.

# B From Protein to Graph Representation

## B.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 $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji} \neq 0$.

## B.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 $\alpha$-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 [14]. Since active sites of proteins are often located on their surfaces, SASA is regarded as a 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 $\alpha$-carbon in each residue to record their 3D position $\boldsymbol{X}_{\text{pos}} \in \mathbb{R}^3$.
- **surface-aware node features**: we follow [12] 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 [20] $\{\sin,\cos\}\circ\{\phi,\psi\}$ from the backbone atom positions. For a specific $v_i$ of the $i$th 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.

### B.3 Edge Attributes

The edge attributes $\boldsymbol{E}\in\mathbb{R}^{92}$ are featured on the connected edges, including

- **positional encoding** represents each node $v_i$ by their sequence number $s_i$. Then, the sequence distance between two nodes $v_i$ and $v_j$ is $d_{i,j}=\min(|s_i-s_j|,65)$. We calculate the threshold number $64$ by plotting the sequence distance distribution and then one-hot encoding of sequence distance *one-hot*$(d_{i,j})\in\mathbb{R}^{65}$ [21]. The last contact signal [20] describes if the two residues contact in the space.
- **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

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

  In aligned with [12], 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.

## C Deep Mutational Scanning Benchmark

In order to validate the performance of our method on zero-shot mutant effect prediction, we test the pre-trained methods on a diverse set of proteins from deep mutational scanning (DMS) experiments,

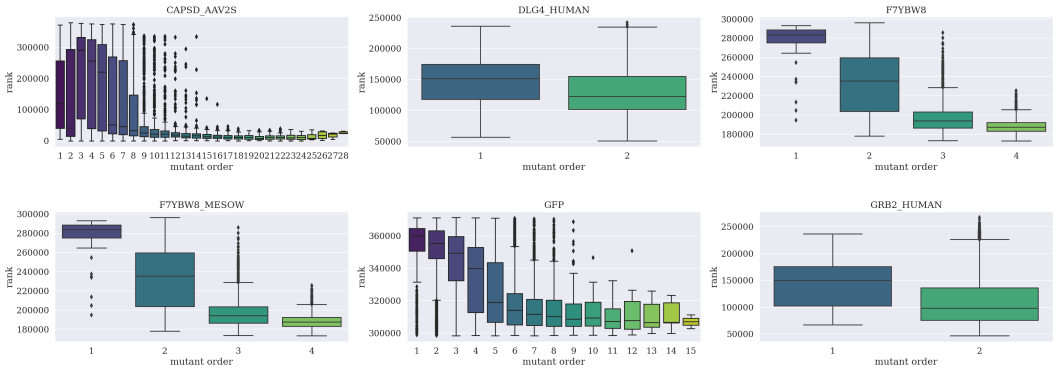


Figure 4: Rank distribution on higher-order mutants. Generally, mutate on more sites would result in a higher score, i.e., a smaller rank value.

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.

### C.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-sites 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 [32]'s work; **GFP** by [36]; **CAPSD_AAV2S**, **DLG4_HUMAN**, and **GRB2_HUMAN** in [27].

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 [17] 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 1 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 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 4, 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.

### C.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 [25, 22]. When the higher-

Table 1: 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 |

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

$$\sum_{t \in T} \log p(\boldsymbol{x}_{\text{aa}} = \hat{\boldsymbol{x}}_{\text{aa}}^{\text{mutant}}) - \log p(\boldsymbol{x}_{\text{aa}} = \boldsymbol{x}_{\text{aa}}^{\text{wild}}), \tag{4}$$

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 specific 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.

# D Model Comparison

In Table 2, we compare the model size and the required resources with the baseline methods: MSA TRANSFORMER [31], ESM-1V [25], and ESM-IF1 [16]. To be specific, the attributes for training the model are provided by the authors, and the inference speed and memory are tested by us. 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 1 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 [25]. The original work by [31] only reports that they used $32\times$V100 GPUs for training, without revealing the training time.

Table 2: Characteristics of different models used in the experiment.

| **model** | MSA TRANSFORMER | ESM-1V | ESM-IF1 | LGN(ours) |
|---|---|---|---|---|
| input | sequence | sequence | sequence+structure | structure |
| training dataset | **Uniref50** (2018-03) | **Uniref90** | **CATH+AF2** (2020-03) | **CATH** v4.3.0 |
| training size | 45M | 98M | 12M | 0.03M |
| max. input token | $1,024$ | $1,024$ | $1,024$ | $2,687$ |
| # parameters | 100M | 650M | 142M | 1.5M |
| # layers | 12 | - | 20 | 6 |
| # head | 12 | - | 8 | - |
| # hid. dim. | - | - | $512-2,048$ | 512 |
| speed (training day) | $13^2$ | 6 | 653 | 0.17 |
| resource (train) | $128\times$V100 | $64\times$V100 | $32\times$V100 | $1\times$3090 |
| inference speed (second) (**GFP**) | $>500$ | 74.84 | 101.99 | 25.28 |

# E Training Details

## E.1 Dataset for Pre-training

**CATH** [29] 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).

We then transform each sample protein sequence of the revised **CATH** dataset into a protein graph, as is defined in Appendix B. We summarize the main properties of the **CATH** dataset in the first three lines of Table 3, 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. 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. 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.

## E.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 have uploaded our model to an anonymous GitHub page at `https://anonymous.4open.science/r/mutation-2486` In addition, 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`

The EGC layers are implemented with the official PyTorch implementation at `https://github.com/lucidrains/egnn-pytorch`.

Table 3: 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 $k$s (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** | $31,848$ | $34$ | $92$ | $8$ | $1,201$ | $150.92$ | $9.96$ | $56$ | $12,004$ | $1,503.62$ |
| **CATH-s40-k5** | $31,848$ | $34$ | $92$ | $8$ | $1,201$ | $150.92$ | $4.98$ | $38$ | $6,004$ | $751.99$ |
| **CATH-s40-k20** | $31,848$ | $34$ | $92$ | $8$ | $1,201$ | $150.92$ | $19.92$ | $56$ | $24,009$ | $3,006.31$ |
| **Mutant-Nsite-k10** | $15$ | $34$ | $92$ | $92$ | $734$ | $365.73$ | $9.99$ | $916$ | $7,333$ | $3,652.27$ |
| **Mutant-Nsite-k5** | $15$ | $34$ | $92$ | $92$ | $734$ | $365.73$ | $4.99$ | $457$ | $3,668$ | $1,826.27$ |
| **Mutant-Nsite-k20** | $15$ | $34$ | $92$ | $92$ | $734$ | $365.73$ | $19.94$ | $1,830$ | $14,656$ | $7,293.93$ |

**Hyper-parameters Setting** The model architecture stacks with $6$ EGC layers and a linear classifier to make predictions. We use ADAM [18] optimizer to optimize our model without 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 decays to $0.0001$. We use gradient clipping equal to $4$ in order to stabilize the training procedure.

# F 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 4. Here we fix the confidence level $p = 0.6$ and the penalty weight $\lambda = 0.2$ for all the losses, except for dihedral loss, for which $\lambda$ is set to $0.5$. The influence of selecting different $p$s will be discussed in the next section.

Table 4: 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 | $0.1752$ | $0.3657$ | $0.1709$ | $0.3345$ |

# G 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 $p$s to guide an empirical choice of it.

Table 5: 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 $\lambda$ 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 |

### G.1 Noisy Ratio on the AA Type

Figure 5-6 demonstrates different perturbation levels on the AA type with wild-type noises and BLOSUM matrix. In Figure 5, 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 6 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 $\lambda$s 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.

### G.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 $p$s to avoid drastic perturbation rates. The influence on the different choices on $p$s 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 $\rho$ on different variants in Table 5 (single-site mutants) and Table 6 (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.

## H 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 $\lambda$(s) should be set as the default value(s)?

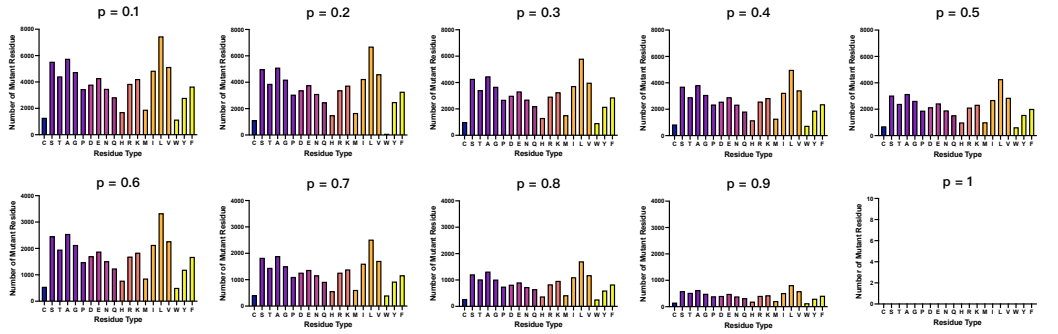


Figure 5: Perturbed label distribution of the training protein graphs at different confidence levels $p$. A higher confidence level results in fewer AAs to observe noisy labels.

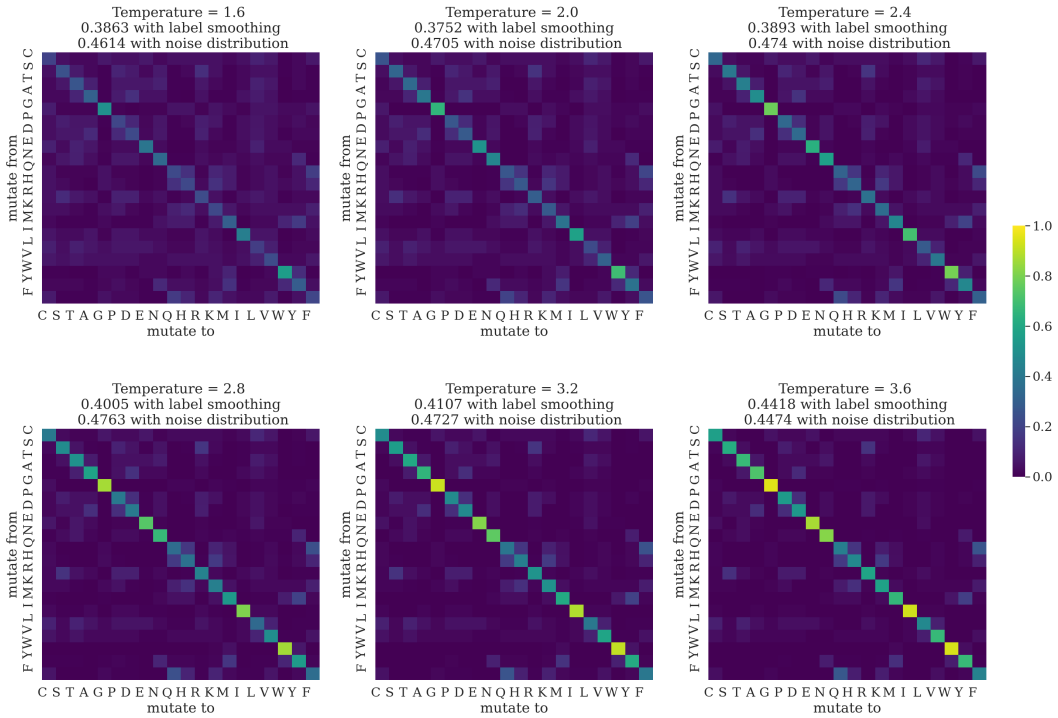


Figure 6: Perturbed label distribution of the training protein graphs at different temperatures. As the BLOSUM matrix has been applied to perturbation distribution and label smoothing of AA types, we report the average Spearman's correlation on the titles of sub-figures over 5 repetitive runs.

Table 6: Average Spearman's rank coefficients at different confidence levels $p$ with different types of probability distribution on **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 $\lambda$ 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 |

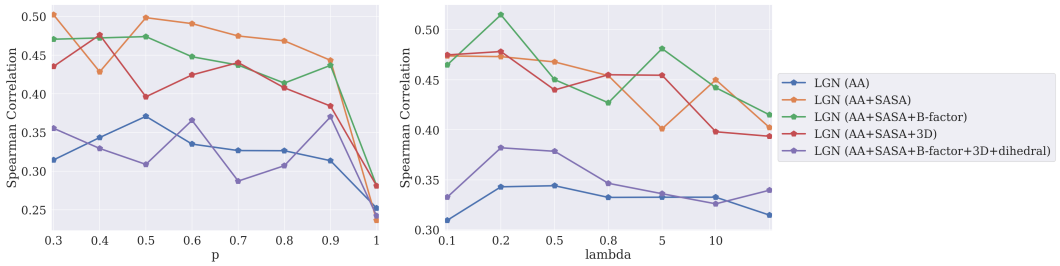


Figure 7: Average Performance on different Choices of $p$s (left) and $\lambda$s (right) with different variants of auxiliary tasks.

### H.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 7 and Table 6-7 in the main table with different $p$s and $\lambda$s and compare them with baseline methods. As a default choice, predicting AA type, SASA and B-factor help generate representative protein graph embedding for further tasks.

### H.2 Choices of Loss Weight

We next detail the influence of the loss function to the overall model performance by selecting different $\lambda$s. 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 7.

Table 7: Test Performance of (model) with different $\lambda$s 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 |

### H.3 Visualized Summary

In detail, Figure 7 visualizes a selection of model performance on the Spearman's correlation with variant $p$s and $\lambda$s 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 $p$s to avoid drastic perturbation rates and search for the optimal $p \in \{0.3, 0.4, \ldots, 0.9, 1\}$. The different choices on $p$s are validated with different learning tasks on the left side of Figure 7 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. More results on different types of perturbation noise and proteins are prepared in Table 6.

We also investigate a wide range of the choices of $\lambda$s. 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 right plot of Figure 7, which demonstrates a relatively flat and steady trend with a mild peak at $\lambda = 0.2, 0.5$. We supplement additional experimental results in Table 7 with different model setups.