Protein Representation Learning by Geometric Structure Pretraining

Anonymous Authors[1]

Abstract

Learning effective protein representations is critical in a variety of tasks in biology such as predicting protein function or structure. Existing approaches usually pretrain protein language models on a large number of unlabeled amino acid sequences and then finetune the models with some labeled data in downstream tasks. Despite the effectiveness of sequence-based approaches, the power of pretraining on known protein structures, which are available in smaller numbers only, has not been explored for protein property prediction, though protein structures are known to be determinants of protein function. In this paper, we propose to pretrain protein representations according to their 3D structures. We first present a simple yet effective encoder to learn the geometric features of a protein. We pretrain the protein graph encoder by leveraging multiview contrastive learning and different self-prediction tasks. Experimental results on both function prediction and fold classification tasks show that our proposed pretraining methods outperform or are on par with the state-of-the-art sequence-based methods, while using much less data. All codes and models will be published upon acceptance.

1. Introduction

Proteins are workhorses of the cell and are implicated in a broad range of applications ranging from therapeutics to material. They consist of a linear chain of amino acids (residues) folding into a specific 3D conformation. As the function annotation of a new protein sequence remains costly and time-consuming, accurate and efficient in silico protein function annotation methods are needed to bridge the existing sequence-function gap. However, existing sequence-based methods neither directly capture nor leverage the available protein structural information that is known to be the determinants of protein functions (Bepler & Berger, 2019; 2021; Rives et al., 2021).

To better utilize this critical structural information, we first propose a simple yet effective structure-based encoder called **GeomEtry-Aware Relational Graph Neural Network (GearNet)**, which encodes spatial information by relational and edge message passing on protein residue graphs. We further introduce five geometric pretraining methods to learn the protein structure encoder. These methods follow popular self-supervised learning frameworks such as contrastive learning (Chen et al., 2020) and self-prediction (Devlin et al., 2018). For contrastive learning, we aim to maximize the similarity between representations of different augmented views from the same protein, while minimizing the similarity between those from different proteins. For self-prediction, the model performs masked prediction of different geometric or biochemical properties, such as residue types, Euclidean distances, angles and dihedral angles.

Extensive experiments on several benchmarks verify our GearNet augmented with edge message passing can consistently outperform existing protein encoders on most tasks in a supervised setting. Further, by employing the proposed pretraining methods, our model trained on less than a million samples achieves comparable or even better results than the state-of-the-art sequence-based encoders pretrained on million- or billion-scale datasets.

2. Structure-Based Protein Encoder

Existing protein encoders are either designed for specific tasks or cumbersome for pretraining due to the dependency on computationally expensive convolutions. Here we propose a simple yet effective protein structure encoder, named *GeomEtry-Aware Relational Graph Neural Network (GearNet)*, which is further enhanced with *edge message passing*.

2.1. Geometry-Aware Relational Graph Neural Network

Given protein structures, our model aims to learn representations encoding their spatial and chemical information. These representations should be invariant under translations and rotations in 3D space. To achieve this requirement, we first construct our protein graph based on the spatial features

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the First Workshop of Pretraining: Perspectives, Pitfalls, and Paths Forward at ICML 2022. Do not distribute.

invariant under these transformations.

Protein graph construction. We represent the structure of a protein as a residue-level relational graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{V}$ and $\mathcal{E}$ denotes the set of nodes and edges respectively, and $\mathcal{R}$ is the set of edge types. We use (i, j, r) to denote the edge from node i to node j with type r. We use n and m to denote the number of nodes and edges, respectively. In this work, each node in the protein graph represents the alpha carbon of a residue with the 3D coordinates of all nodes $\boldsymbol{x} \in \mathbb{R}^{n\times 3}$.

Then, we add three different types of edges into our graphs: sequential edges, radius edges and K-nearest neighbor edges. Among these, sequential edges will be further divided into 5 types of edges based on the relative sequential distance $d \in \{-2, -1, 0, 1, 2\}$ between two end nodes, where we add sequential edges only between the nodes within the sequential distance of 2. These edge types reflect different geometric properties, which all together yield a comprehensive featurization of proteins. More details of graph construction can be found in Appendix C.1.

Relational graph convolutional layer. To balance model capacity and memory cost, we use a relational graph convolutional neural network (Schlichtkrull et al., 2018) to learn graph representations, where a convolutional kernel matrix is shared within each edge type and there are $|\mathcal{R}|$ different kernel matrices in total. Formally, the relational graph convolutional layer used in our model is defined as

$$\boldsymbol{h}_i^{(l)} = \boldsymbol{h}_i^{(l-1)} + \sigma\left(\mathrm{BN}\left(\sum_{r\in\mathcal{R}} \boldsymbol{W}_r \sum_{j\in\mathcal{N}_r(i)} \boldsymbol{h}_j^{(l-1)}\right)\right).$$

Specifically, we use node features $\boldsymbol{f}_i$ as initial representations $\boldsymbol{h}_i^{(0)}$. Then, given the node representation $\boldsymbol{h}_i^{(l)}$ for node i at the l-th layer, we compute updated node representation by aggregating features from neighboring nodes $\mathcal{N}_r(i)$, where $\mathcal{N}_r(i) = \{j \in \mathcal{V} | (j, i, r) \in \mathcal{E}\}$ denotes the neighborhood of node i with the edge type r, and $\boldsymbol{W}_r$ denotes the learnable convolutional kernel matrix for edge type r. Here BN denotes a batch normalization layer and we use a ReLU function as the activation $\sigma(\cdot)$. Finally, we update $\boldsymbol{h}_i^{(l)}$ with a residual connection from last layer.

2.2. Edge Message Passing Layer

Inspired by models on small molecules (Klicpera et al., 2020), we propose a variant of GearNet enhanced with an edge message passing layer, named as **GearNet-Edge**.

We first construct a relational graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{R}')$ among edges, also known as *line graph* in the literature (Harary & Norman, 1960). Each node in the graph $\mathcal{G}'$ corresponds to an edge in the original graph. $\mathcal{G}'$ links edge (i, j, r_1) to (w, k, r_2) if and only if $j = w$ and $i \neq k$. The type of this edge is determined by the angle between (i, j, r_1) and (w, k, r_2). We discretize the range $[0, \pi]$ into 8 bins and use the index of the bin as the edge type.

Then, we apply a similar relational graph convolutional network on the graph $\mathcal{G}'$ for the message function. Formally, the edge message passing layer is defined as

$$\boldsymbol{m}_{(i,j,r_1)}^{(l)} = \sigma\left(\mathrm{BN}\left(\sum_{r\in\mathcal{R}'} \boldsymbol{W}_r' \sum_{\substack{(w,k,r_2)\in \\ \mathcal{N}_r'((i,j,r_1))}} \boldsymbol{m}_{(w,k,r_2)}^{(l-1)}\right)\right).$$

Here we use $\boldsymbol{m}_{(i,j,r_1)}^{(l)}$ to denote the message function for edge (i, j, r_1) in the l-th layer with $\boldsymbol{m}_{(i,j,r_1)}^{(0)} = \boldsymbol{f}_{(i,j,r_1)}$. Similarly, the message function for edge (i, j, r_1) will be updated by aggregating features from its neighbors $\mathcal{N}_r'((i, j, r_1))$, where $\mathcal{N}_r'((i, j, r_1)) = \{(w, k, r_2) \in \mathcal{V}' | ((w, k, r_2), (i, j, r_1), r) \in \mathcal{E}'\}$ denotes the set of incoming edges of (i, j, r_1) with relation type r in graph $\mathcal{G}'$.

Finally, we replace the update function in the original graph with the following one:

$$\boldsymbol{h}_i^{(l)} = \boldsymbol{h}_i^{(l-1)} + \sigma\left(\mathrm{BN}\left(\textstyle\sum_{r\in\mathcal{R}} \boldsymbol{W}_r \sum_{j\in\mathcal{N}_r(i)} (\boldsymbol{h}_j^{(l-1)} + \mathrm{FC}(\boldsymbol{m}_{(j,i,r)}^{(l)}))\right)\right),$$

where $\mathrm{FC}(\cdot)$ denotes a linear layer over the message.

Notably, it is novel to use relational message passing to model spatial interactions among residues. In addition, to the best of our knowledge, this is the first work that explores edge message passing for protein representation learning.

3. Geometric Pretraining Methods

In this section, we study how to leverge massive unlabeled protein structures to boost protein representation learning. We follow two popular self-supervised learning frameworks: contrastive learning and self-prediction, and propose five different pretraining methods for structure-based encoders.

3.1. Multiview Contrastive Learning

Inspired by recent contrastive learning methods (Oord et al., 2018; Chen et al., 2020), our framework learns representations by maximizing the similarity between representations of different augmented views of the same protein while minimizing the agreement between views of different proteins.

More specifically, given a protein graph $\mathcal{G}$, we first sample two different views $\mathcal{G}_x$ and $\mathcal{G}_y$ via a stochastic augmentation module. We then compute the graph representations $\boldsymbol{h}_x$ and $\boldsymbol{h}_y$ of two views using our structure-based encoder. Following SimCLR (Chen et al., 2020), a two-layer MLP projection head is further applied to map the representations to a lower-dimensional space, denoted as $\boldsymbol{z}_x$ and $\boldsymbol{z}_y$. Finally, a contrastive loss function is defined by distinguishing

views from the same or different proteins using their similarities. For a positive pair x and y, we treat views from other proteins in the same mini-batch as negative pairs. Mathematically, the loss function for a positive pair of views x and y can be written as:

$$\mathcal{L}_{x,y} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_x, \boldsymbol{z}_y)/\tau)}{\sum_{k=1}^{2B} \mathbb{1}_{[k\neq x]} \exp(\text{sim}(\boldsymbol{z}_y, \boldsymbol{z}_k)/\tau)}, \quad (1)$$

where B denotes the batch size, τ denotes the temperature, $\mathbb{1}_{[k\neq x]} \in \{0, 1\}$ is an indicator function that is equal to 1 if and only if $k \neq x$. And the similarity function $\text{sim}(\boldsymbol{u}, \boldsymbol{v})$ is defined by the cosine similarity between $\boldsymbol{u}$ and $\boldsymbol{v}$.

Data augmentation. As suggested in (Chen et al., 2020), data augmentation schemes play an important role in contrastive learning. In this work, we propose a principled way to generate diverse views of a protein.

We randomly choose a *cropping* function to make the protein graph smaller so that larger-size batches can be used for pretraining. Here we consider two cropping functions:

- **Subsequence:** randomly sample a consecutive segment of protein sequences and take the corresponding subgraph from the protein residue graph.
- **Subspace:** randomly sample a residue as the center and select all residues within a Euclidean ball with a predefined radius.

Then, we randomly choose one of the following two transformations with equal probability to apply on the cropped protein graphs.

- **Identity:** no transformation.
- **Random edge masking:** randomly mask edges with a fixed mask rate equal to 0.15.

3.2. Self-Prediction Methods

Another line of research is based on the recent progress of self-prediction methods in natural language processing (Devlin et al., 2018; Brown et al., 2020). Given a protein, our objective is to predict one part of the protein given the remainder of the structure. Here, we propose four self-supervised tasks based on geometric or biochemical properties.

Residue Type Prediction. Our first method is based on the masked language modeling objective, which has been widely used in pretraining large-scale protein language models (Bepler & Berger, 2021). For each protein, we randomly mask node features of some residues and then let structure-based encoders to predict these masked residue types. For a masked node i, the learning objective is defined as:

$$\mathcal{L}_i = \text{CE}(f_{\text{residue}}(\boldsymbol{h}'_i), \boldsymbol{f}_i),$$

where $\boldsymbol{h}'_i$ denotes the representations of node i after masking, $f_{\text{residue}}(\cdot)$ is an MLP classification head and $\text{CE}(\cdot, \cdot)$ denotes the cross entropy loss. This method is also known as Attribute Masking in the literature of pretraining on small molecules (Hu et al., 2019).

Distance Prediction. In order to learn local spatial structures, we use our learned representations to predict the Euclidean distance between two nodes connected in the protein graph. First, we randomly select a fixed number of edges from the original graph. Then, these edges will be removed when feeding the graph into the encoder. For a selected edge, the representations of its two end nodes will be concatenated to predict the distance between them. More concretely, the loss function for an edge (i, j, r) will be defined as:

$$\mathcal{L}_{(i,j,r)} = (f_{\text{dist}}(\boldsymbol{h}'_i, \boldsymbol{h}'_j) - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2)^2,$$

where $f_{\text{dist}}(\cdot)$ is an MLP prediction head.

Angle Prediction. Besides distances, angles between edges are also important features that reflect the relative position between residues. Similarly, we can define a masked geometric loss by 1) randomly selecting two adjacent edges, 2) removing these edges from the graph, 3) using the three end nodes of the pair of edges to predict the angle between them. Here we discretized the angles by cutting the range $[0, \pi]$ into 8 bins. The objective for a selected pair of edges (i, j, r_1) and (j, k, r_2) is to predict which bin the angle between them will belong to:

$$\mathcal{L}_{(i,j,r_1),(j,k,r_2)} = \text{CE}(f_{\text{angle}}(\boldsymbol{h}'_i, \boldsymbol{h}'_j, \boldsymbol{h}'_k), \text{bin}(\angle ijk)),$$

where $f_{\text{angle}}(\cdot)$ is an MLP classification head and $\text{bin}(\cdot)$ is used to discretize the angle.

Dihedral Prediction. As shown in (Klicpera et al., 2021), the dihedral angles between three edges can provide important clues about the relative directional information. Therefore, we can also construct a masked geometric objective by predicting the dihedral angle between three consecutive edges (i, j, r_1), (j, k, r_2) and (k, t, r_3):

$$\mathcal{L}_{(i,j,r_1),(j,k,r_2),(k,t,r_3)} = \text{CE}(f(\boldsymbol{h}'_i, \boldsymbol{h}'_j, \boldsymbol{h}'_k, \boldsymbol{h}'_t), \text{bin}(\angle ijkt)),$$

where $f(\cdot)$ is an MLP classification head and $\text{bin}(\cdot)$ is used to discretize the dihedral angles.

4. Experiments

In this section, we conduct experiments on four downstream tasks, where the experimental setups are described in Append. E.1 and the results are reported in Table 1, including all models with and without pretraining. The following conclusions can be drawn from the results:

Table 1: F_{max} on EC and GO prediction and Accuracy (%) on fold and reaction classification. [†] denotes results taken from (Wang et al., 2022) and [*] denotes results taken from (Hermosilla et al., 2021). For pretraining, we select the model with the best performance when training from scratch, *i.e.*, GearNet-Edge for EC, GO, Reaction and GearNet-Edge-IEConv for Fold Classification. We omit the model name and use the pretraining methods to name our pretrained models.

	Method	Pretraining Dataset (Size)	EC	GO			Fold Classification				Reaction
				BP	MF	CC	Fold	Super.	Fam.	Avg.	
w/o pretraining	CNN (Shanehsazzadeh et al., 2020)	-	0.545	0.244	0.354	0.287	11.3	13.4	53.4	26.0	51.7
	ResNet (Rao et al., 2019)	-	0.605	0.280	0.405	0.304	10.1	7.21	23.5	13.6	24.1
	LSTM (Rao et al., 2019)	-	0.425	0.225	0.321	0.283	6.41	4.33	18.1	9.61	11.0
	Transformer (Rao et al., 2019)	-	0.238	0.264	0.211	0.405	9.22	8.81	40.4	19.4	26.6
	GCN (Kipf & Welling, 2017)	-	0.320	0.252	0.195	0.329	16.8*	21.3*	82.8*	40.3*	67.3*
	GAT (Veličković et al., 2018)	-	0.368	0.284†	0.317†	0.385†	12.4	16.5	72.7	33.8	55.6
	GVP (Jing et al., 2021)	-	0.489	0.326†	0.426†	0.420†	16.0	22.5	83.8	40.7	65.5
	3DCNN_MQA (Derevyanko et al., 2018)	-	0.077	0.240	0.147	0.305	31.6*	45.4*	92.5*	56.5*	72.2*
	GraphQA (Baldassarre et al., 2021)	-	0.509	0.308	0.329	0.413	23.7*	32.5*	84.4*	46.9*	60.8*
	New IEConv (Hermosilla & Ropinski, 2022)	-	0.735	0.374	0.544	0.444	47.6*	70.2*	99.2*	72.3*	**87.2***
	GearNet	-	0.730	0.356	0.503	0.414	28.4	42.6	95.3	55.4	79.4
	GearNet-IEConv	-	0.800	0.381	0.563	0.422	42.3	64.1	99.1	68.5	83.7
	GearNet-Edge	-	**0.810**	**0.403**	**0.580**	**0.450**	44.0	66.7	99.1	69.9	86.6
	GearNet-Edge-IEConv	-	**0.810**	**0.400**	**0.581**	0.430	**48.3**	**70.3**	**99.5**	**72.7**	85.3
w/ pretraining	DeepFRI (Gligorijević et al., 2021)	Pfam (10M)	0.631	0.399	0.465	0.460	15.3*	20.6*	73.2*	36.4*	63.3*
	ESM-1b (Rives et al., 2021)	UniRef50 (24M)	0.864	0.470	**0.657**	0.488	26.8	60.1	97.8	61.5	83.1
	ProtBERT-BFD (Elnaggar et al., 2021)	BFD (2.1B)	0.838	0.279†	0.456†	0.408†	26.6*	55.8*	97.6*	60.0*	72.2*
	LM-GVP (Wang et al., 2022)	UniRef100 (216M)	0.664	0.417†	0.545†	**0.527**†	-	-	-	-	-
	New IEConv (Hermosilla & Ropinski, 2022)	PDB (476K)	-	-	-	-	50.3*	**80.6***	99.7*	76.9*	**87.6***
	Multiview Contrast	AlphaFoldDB (805K)	**0.874**	**0.490**	**0.654**	0.488	54.1	**80.5**	**99.9**	**78.1**	**87.5**
	Residue Type Prediction	AlphaFoldDB (805K)	0.843	0.430	0.604	0.465	48.8	71.0	99.4	73.0	86.6
	Distance Prediction	AlphaFoldDB (805K)	0.839	0.448	0.616	0.464	50.9	73.5	99.4	74.6	**87.5**
	Angle Prediction	AlphaFoldDB (805K)	0.853	0.458	0.625	0.473	**56.5**	76.3	99.6	77.4	86.8
	Dihedral Prediction	AlphaFoldDB (805K)	0.859	0.458	0.626	0.465	51.8	77.8	99.6	75.9	87.0

Our structure-based encoders outperform all baselines without pretraining on 7 of 8 datasets. By comparing the first three blocks, we find that the vanilla GearNet can already obtain competitive results against other baselines on three function prediction tasks (EC, GO, Reaction). After adding the edge message passing mechanism, our method GearNet-Edge significantly outperforms other baselines on EC, GO-BP and GO-MF and achieves competitive results on GO-CC. By further adding IEConv (Hermosilla & Ropinski, 2022) layers into our model, the GearNet-Edge-IEConv model can achieve the best results on the fold classification task. These strong performance demonstrates the advantages of our proposed structure-based encoders.

Structure-based encoders benefit a lot from pretraining with unlabeled protein structures. Comparing the results in the third and fifth blocks, it can be observed that models with all five proposed pretraining methods show large improvements over models trained from scratch. Among these methods, Multiview Contrast is the best on 7 of 8 datasets and achieve the state-of-the-art results on EC, GO-BP, GO-MF, Fold and Reaction classification tasks. This proves the effectiveness of our pretraining strategies.

Pretrained structure-based encoders perform on par with or even better than sequence-based encoders pretrained with much more data. The last two blocks show the comparision between pretrained sequence-based and structure-based models. It should be noted that our models are pretrained on a dataset with fewer than one million structures, whereas all sequence-based pretraining baselines are pretrained on million- or billion-scale sequence databases. Though pretrained with an order of magnitude less data, our model can achieve comparable or better results against these sequence-based models. Besides, our model is the only one that can achieve good performance on all four tasks, given that sequence-based models do not perform well on fold classification. This again shows the potential of structure-based pretraining for learning protein representations.

5. Conclusions

In this work, we propose a simple yet effective structure-based encoder for protein representation learning, which performs relational and edge message passing on protein residue graphs. Moreover, five self-supervised pretraining methods are proposed following two standard frameworks: contrastive learning and self-prediction. Comprehensive experiments over multiple tasks verify that our model outperforms previous encoders when trained from scratch and achieve comparable or even better results than the state-of-the-art baselines while pretraining with much less data.

References

Akdel, M., Pires, D. E., Pardo, E. P., Jänes, J., Zalevsky, A. O., Mészáros, B., Bryant, P., Good, L. L., Laskowski, R. A., Pozzati, G., et al. A structural biology community assessment of alphafold 2 applications. *bioRxiv*, 2021.

Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

Baldassarre, F., Menéndez Hurtado, D., Elofsson, A., and Azizpour, H. Graphqa: protein model quality assessment using graph convolutional networks. *Bioinformatics*, 37 (3):360–366, 2021.

Bartók, A. P., Payne, M. C., Kondor, R., and Csányi, G. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Physical review letters*, 104(13):136403, 2010.

Bartók, A. P., Kondor, R., and Csányi, G. On representing chemical environments. *Physical Review B*, 87(18): 184115, 2013.

Behler, J. and Parrinello, M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical review letters*, 98(14):146401, 2007.

Bepler, T. and Berger, B. Learning protein sequence embeddings using information from structure. *arXiv preprint arXiv:1902.08661*, 2019.

Bepler, T. and Berger, B. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 12(6): 654–669, 2021.

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

Biswas, S., Khimulya, G., Alley, E. C., Esvelt, K. M., and Church, G. M. Low-n protein engineering with data-efficient deep learning. *Nature Methods*, 18(4):389–396, 2021.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Chen, C., Ye, W., Zuo, Y., Zheng, C., and Ong, S. P. Graph networks as a universal machine learning framework for molecules and crystals. *Chemistry of Materials*, 31(9): 3564–3572, 2019.

Chen, C., Zhou, J., Wang, F., Liu, X., and Dou, D. Structure-aware protein self-supervised learning. *arXiv preprint arXiv:2204.04213*, 2022.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Chmiela, S., Tkatchenko, A., Sauceda, H. E., Poltavsky, I., Schütt, K. T., and Müller, K.-R. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

Consortium, T. U. Uniprot: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, 2021.

Dai, B. and Bailey-Kellogg, C. Protein interaction interface region prediction by geometric deep learning. *Bioinformatics*, 2021.

Dallago, C., Mou, J., Johnston, K. E., Wittmann, B., Bhattacharya, N., Goldman, S., Madani, A., and Yang, K. K. Flip: Benchmark tasks in fitness landscape inference for proteins. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Derevyanko, G., Grudinin, S., Bengio, Y., and Lamoureux, G. Deep convolutional networks for quality assessment of protein folds. *Bioinformatics*, 34(23):4046–4053, 2018.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Yu, W., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., and Rost, B. 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.

Gainza, P., Sverrisson, F., Monti, F., Rodola, E., Boscaini, D., Bronstein, M., and Correia, B. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17(2):184–192, 2020.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Gligorijević, V., Renfrew, P. D., Kosciolek, T., Leman, J. K., Berenberg, D., Vatanen, T., Chandler, C., Taylor, B. C., Fisk, I. M., Vlamakis, H., et al. Structure-based protein function prediction using graph convolutional networks. *Nature communications*, 12(1):1–14, 2021.

Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

Harary, F. and Norman, R. Z. Some properties of line digraphs. *Rendiconti del circolo matematico di palermo*, 9(2):161–168, 1960.

Hassani, K. and Khasahmadi, A. H. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pp. 4116–4126. PMLR, 2020.

He, L., Zhang, S., Wu, L., Xia, H., Ju, F., Zhang, H., Liu, S., Xia, Y., Zhu, J., Deng, P., et al. Pre-training co-evolutionary protein representation via a pairwise masked language model. *arXiv preprint arXiv:2110.15527*, 2021.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Hermosilla, P. and Ropinski, T. Contrastive representation learning for 3d protein structures. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=VINWzIM6_6.

Hermosilla, P., Schäfer, M., Lang, M., Fackelmann, G., Vázquez, P. P., Kozlíková, B., Krone, M., Ritschel, T., and Ropinski, T. Intrinsic-extrinsic convolution and pooling for learning on 3d protein structures. *International Conference on Learning Representations*, 2021.

Hou, J., Adhikari, B., and Cheng, J. Deepsf: deep convolutional neural network for mapping protein sequences to folds. *Bioinformatics*, 34(8):1295–1303, 2018.

Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., and Leskovec, J. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. Generative models for graph-based protein design. *Advances in Neural Information Processing Systems*, 32:15820–15831, 2019.

Jarzab, A., Kurzawa, N., Hopf, T., Moerch, M., Zecha, J., Leijten, N., Bian, Y., Musiol, E., Maschberger, M., Stoehr, G., et al. Meltome atlas—thermal proteome stability across the tree of life. *Nature methods*, 17(5):495–503, 2020.

Jing, B., Eismann, S., Soni, P. N., and Dror, R. O. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=1YLJDvSx6J4.

Jørgensen, P. B., Jacobsen, K. W., and Schmidt, M. N. Neural message passing with edge updates for predicting properties of molecules and materials. *arXiv preprint arXiv:1806.03146*, 2018.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Kipf, T. N. and Welling, M. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Klicpera, J., Groß, J., and Günnemann, S. Directional message passing for molecular graphs. In *International Conference on Learning Representations (ICLR)*, 2020.

Klicpera, J., Becker, F., and Günnemann, S. Gemnet: Universal directional graph neural networks for molecules. *arXiv preprint arXiv:2106.08903*, 2021.

Liu, Y., Wang, L., Liu, M., Zhang, X., Oztekin, B., and Ji, S. Spherical message passing for 3d graph networks. *arXiv preprint arXiv:2102.05013*, 2021.

Lu, A. X., Zhang, H., Ghassemi, M., and Moses, A. M. Self-supervised contrastive learning of protein representations by mutual information maximization. *BioRxiv*, 2020.

McInnes, L., Healy, J., and Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T., and Rives, A. Language models enable zero-shot prediction of the effects of mutations on protein function. *bioRxiv*, 2021.

Mirdita, M., Schütze, K., Moriwaki, Y., Heo, L., Ovchinnikov, S., and Steinegger, M. Colabfold - making protein folding accessible to all. *bioRxiv*, 2021. doi: 10.1101/2021.08.15.456425. URL https://www.biorxiv.org/content/early/2021/10/29/2021.08.15.456425.

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

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Qiu, J., Chen, Q., Dong, Y., Zhang, J., Yang, H., Ding, M., Wang, K., and Tang, J. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1150–1160, 2020.

Radivojac, P., Clark, W. T., Oron, T. R., Schnoes, A. M., Wittkop, T., Sokolov, A., Graim, K., Funk, C., Verspoor, K., Ben-Hur, A., et al. A large-scale evaluation of computational protein function prediction. *Nature methods*, 10(3):221–227, 2013.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y. S. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, 2019.

Rao, R., Liu, J., Verkuil, R., Meier, J., Canny, J. F., Abbeel, P., Sercu, T., and Rives, A. Msa transformer. *bioRxiv*, 2021.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., 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), 2021.

Rocklin, G. J., Chidyausiku, T. M., Goreshnik, I., Ford, A., Houliston, S., Lemak, A., Carter, L., Ravichandran, R., Mulligan, V. K., Chevalier, A., et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357(6347):168–175, 2017.

Rong, Y., Bian, Y., Xu, T., Xie, W., Wei, Y., Huang, W., and Huang, J. Self-supervised graph transformer on large-scale molecular data. *arXiv preprint arXiv:2007.02835*, 2020.

Sarkisyan, K. S., Bolotin, D. A., Meer, M. V., Usmanova, D. R., Mishin, A. S., Sharonov, G. V., Ivankov, D. N., Bozhanova, N. G., Baranov, M. S., Soylemez, O., et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, 2016.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pp. 593–607. Springer, 2018.

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8 (1):1–8, 2017a.

Schütt, K. T., Kindermans, P.-J., Sauceda, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *arXiv preprint arXiv:1706.08566*, 2017b.

Shanehsazzadeh, A., Belanger, D., and Dohan, D. Is transfer learning necessary for protein landscape prediction? *arXiv preprint arXiv:2011.03443*, 2020.

Somnath, V. R., Bunne, C., and Krause, A. Multi-scale representation learning on proteins. *Advances in Neural Information Processing Systems*, 34, 2021.

Steinegger, M. and Söding, J. Clustering huge protein sequence sets in linear time. *Nature communications*, 9(1): 1–8, 2018.

Steinegger, M. and Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, Nov 2017. ISSN 1546-1696. doi: 10.1038/nbt.3988.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pp. 3319–3328. PMLR, 2017.

Suzek, B. E., Huang, H., McGarvey, P., Mazumder, R., and Wu, C. H. Uniref: comprehensive and non-redundant uniprot reference clusters. *Bioinformatics*, 23(10):1282–1288, 2007.

Sverrisson, F., Feydy, J., Correia, B. E., and Bronstein, M. M. Fast end-to-end learning on protein surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15272–15281, 2021.

Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., et al. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 2021.

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

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. accepted as poster.

Wang, Y., You, Z.-H., Yang, S., Li, X., Jiang, T.-H., and Zhou, X. A high efficient biological language model for predicting protein–protein interactions. *Cells*, 8(2):122, 2019.

Wang, Z., Combs, S. A., Brand, R., Calvo, M. R., Xu, P., Price, G., Golovach, N., Salawu, E. O., Wise, C. J., Ponnapalli, S. P., et al. Lm-gvp: an extensible sequence and structure informed deep learning framework for protein property prediction. *Scientific reports*, 12(1):1–12, 2022.

Webb, O. F., Phelps, T. J., Bienkowski, P. R., Digrazia, P. M., White, D. C., and Sayler, G. S. Enzyme nomenclature. 1992.

Wu, N. C., Dai, L., Olson, C. A., Lloyd-Smith, J. O., and Sun, R. Adaptation in protein fitness landscapes is facilitated by indirect paths. *Elife*, 5:e16965, 2016.

Xu, M., Wang, H., Ni, B., Guo, H., and Tang, J. Self-supervised graph-level representation learning with local and global structure. *arXiv preprint arXiv:2106.04113*, 2021.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33: 5812–5823, 2020.

Zhu, Z., Shi, C., Zhang, Z., Liu, S., Xu, M., Yuan, X., Zhang, Y., Chen, J., Cai, H., Lu, J., et al. Torchdrug: A powerful and flexible machine learning platform for drug discovery. *arXiv preprint arXiv:2202.08320*, 2022.

A. Related Work

Previous works seek to learn protein representations based on different modalities of proteins, including amino acid sequences (Rao et al., 2019; Elnaggar et al., 2021; Rives et al., 2021), multiple sequence alignments (MSAs) (Rao et al., 2021; Biswas et al., 2021; Meier et al., 2021) and protein structures (Hermosilla et al., 2021; Gligorijević et al., 2021; Somnath et al., 2021). These works share the common goal of learning informative protein representations that can benefit various downstream applications, like predicting protein function (Gligorijević et al., 2021; Rives et al., 2021) and protein-protein interaction (Wang et al., 2019), as well as designing protein sequences (Biswas et al., 2021).

A.1. Sequence-Based Methods

Sequence-based protein representation learning is mainly inspired by the methods of modeling natural language sequences. Recent methods aim to capture the biochemical and co-evolutionary knowledge underlying a large-scale protein sequence corpus by self-supervised pretraining, and such knowledge is then transferred to specific downstream tasks by finetuning. Typical pretraining objectives explored in existing methods include next amino acid prediction (Alley et al., 2019; Elnaggar et al., 2021), masked language modeling (MLM) (Rao et al., 2019; Elnaggar et al., 2021; Rives et al., 2021), pairwise MLM (He et al., 2021) and contrastive predictive coding (CPC) (Lu et al., 2020). Compared to sequence-based approaches that learn in the whole sequence space, MSA-based methods (Rao et al., 2021; Biswas et al., 2021; Meier et al., 2021) leverage the sequences within a protein family to capture the conserved and variable regions of homologous sequences, which imply specific structures and functions of the protein family.

A.2. Structure-Based Methods

Although sequence-based methods pretrained on large-scale databases perform well, structure-based methods should be, in principle, a better solution to learning an informative protein representation, as the function of a protein is determined by its structure. This line of works seeks to encode spatial information in protein structures by 3D CNNs (Derevyanko et al., 2018) or graph neural networks (GNNs) (Gligorijević et al., 2021; Baldassarre et al., 2021; Jing et al., 2021). Among these methods, IEConv (Hermosilla et al., 2021) tries to fit the inductive bias of protein structure modeling, which introduced a graph convolution layer incorporating intrinsic and extrinsic distances between nodes. Another potential direction is to extract features from protein surfaces (Gainza et al., 2020; Sverrisson et al., 2021; Dai & Bailey-Kellogg, 2021). (Somnath et al., 2021) combined the advantages of both worlds and proposed a parameter-efficient multi-scale model. Besides, there are also works that enhance pretrained sequence-based models by incorporating structural information in the pretraining stage (Bepler & Berger, 2021) or finetuning stage (Wang et al., 2022).

Despite progress in the design of structure-based encoders, there are few works focusing on structure-based pretraining for proteins. To the best of our knowledge, the only attempt is two concurrent works (Hermosilla & Ropinski, 2022) and (Chen et al., 2022), which apply contrastive learning and self-prediction methods on a small set of tasks, respectively. Compared with these existing works, our proposed encoder is conceptually simpler and more effective on many different tasks, thanks to the proposed relational graph convolutional layer and edge message passing layer, which are able to efficiently capture both the sequential and structural information. Furthermore, we introduce five structure-based pretraining methods within the contrastive learning and self-prediction frameworks, which can serve as a solid starting point for enabling self-supervised learning on protein structures.

A.3. Structure-based Encoders for Biological Molecules

Following the early efforts (Behler & Parrinello, 2007; Bartók et al., 2010; 2013; Chmiela et al., 2017) of building machine learning systems for molecules by hand-crafted atomic features, recent works exploited end-to-end message passing neural networks (MPNNs) (Gilmer et al., 2017) to encode the structures of small molecules and macromolecules like proteins. Specifically, existing methods employed node/atom message passing (Gilmer et al., 2017; Schütt et al., 2017a;b), edge/bond message passing (Jørgensen et al., 2018; Chen et al., 2019) and directional information (Klicpera et al., 2020; Liu et al., 2021; Klicpera et al., 2021) to encode 2D or 3D molecular graphs.

Compared to small molecules, structural representations of proteins are more diverse, including residue-level graphs, atom-level graphs and protein surfaces. There are some recent models designed to encode residue-level graphs (Hermosilla et al., 2021; Hermosilla & Ropinski, 2022) and protein surfaces (Gainza et al., 2020; Sverrisson et al., 2021), and they achieved impressive results on various tasks. However, these models are either not expressive enough to capture edge interactions or too complicated for representation learning.

A.4. Pretraining Graph Neural Networks

Our work is also related to the recent efforts of pretraining graph neural networks (GNNs), which sought to learn graph representations in a self-supervised fashion. In this domain, various self-supervised pretext tasks, like edge prediction (Kipf & Welling, 2016; Hamilton et al., 2017), context prediction (Hu et al., 2019; Rong et al., 2020), node/edge attribute reconstruction (Hu et al., 2019) and contrastive

learning (Hassani & Khasahmadi, 2020; Qiu et al., 2020; You et al., 2020; Xu et al., 2021), are designed to acquire knowledge from unlabeled graphs. In this work, we focus on learning representations of residue-level graphs of proteins in a self-supervised way. To attain this goal, we design several novel protein-specific pretraining methods to learn the proposed structure-based GNN encoder.

B. Broader Impact

This research project focuses on learning effective protein representations via pretraining with a large number of unlabeled protein structures. Compared to the conventional sequence-based pretraining methods, our approach is able to leverage structural information and thus provide better representations. This merit enables more in-depth analysis of protein research and can potentially benefit many real-world applications, like protein function prediction and sequence design.

However, it cannot be denied that some harmful activities could be augmented by powerful pretrained models, *e.g.*, designing harmful drugs. We expect future studies will mitigate these issues.

C. More Details of GearNet

In this section, we describe more details about the implementation of our GearNet. The whole pipeline of our structure-based encoder is depicted in Figure 2.

C.1. Protein Graph Construction

For graph construction, we use three different ways to add edges:

1. **Sequential edges.** The i-th residue and the j-th residue will be linked by an edge if the sequential distance between them is below a predefined threshold d_{seq}, i.e., $|j - i| < d_{\text{seq}}$. The type of each sequential edge is determined by their relative position $d = j - i$ in the sequence. Hence, there are $2d_{\text{seq}} - 1$ types of sequential edges.
2. **Radius edges.** Following previous works, we also add edges between two nodes i and j when the Euclidean distance between them is smaller than a threshold d_{radius}.
3. **K-nearest neighbor edges.** Since the scales of spatial coordinates may vary among different proteins, a node will be also connected to its k-nearest neighbors based on the Euclidean distance. In this way, the density of spatial edges are guaranteed to be comparable among different protein graphs.

Since we are not interested in spatial edges between residues close with each other in the sequence, we further add a filter to the latter two kinds of edges. Specifically, for a radius or KNN edge connecting the i-th residue and j-th residue, it will be removed if the sequential distance between them is lower than a long range interaction cutoff d_{long}, *i.e.*, $|i - j| < d_{\text{long}}$.

In this paper, we set the sequential distance threshold $d_{\text{seq}} = 3$, the radius $d_{\text{radius}} = 10.0$Å, the number of neighbors $k = 10$ and the long range interaction cutoff $d_{\text{long}} = 5$. By regarding radius edges and KNN edges as two separate edge types, there will be totally $2d_{\text{seq}} + 1 = 7$ different types of edges.

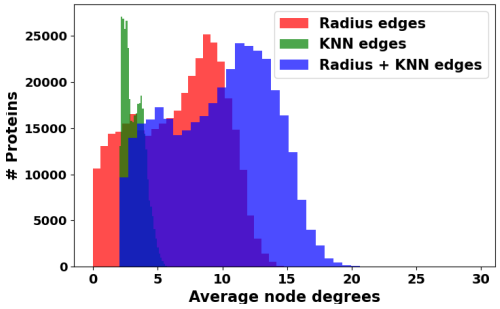

Figure 1: The average degree distribution on AlphaFold Database.

Necessity of spatial edges. Here we explain the necessity of radius and KNN edges by statistics and intuitions. These two kinds of edges result in very different degree distributions. In Figure 1, we plot the average degree distribution over all proteins in AlphaFold Database v1. If we only consider KNN edges, the node degrees in protein graphs are close to a constant, which makes it difficult to capture those areas with dense interactions between residues. If we only consider radius edges, then there will be about 45,000 proteins with average degrees lower than two. In these sparse graphs, pretraining cannot capture structural information effectively, *e.g.*, Angle Prediction with limited edge pairs and Dihedral Prediction with limited edge triplets. Such sparsity can hard be overcome by tuning radius cutoff, for the various scales of average distance on different proteins. By simply combining two kinds of edges, we can overcome these issues.

C.2. Node and edge features.

Most previous structure-based encoders designed for biological molecules (Baldassarre et al., 2021; Hermosilla et al., 2021) used many chemical and spatial features, some of which are difficult to obtain or time-consuming to calculate. In contrast, we only use the one-hot encoding of residue types with one additional dimension for unknown types as node features, denoted as $\boldsymbol{f} \in \{0, 1\}^{n \times 21}$, which is enough

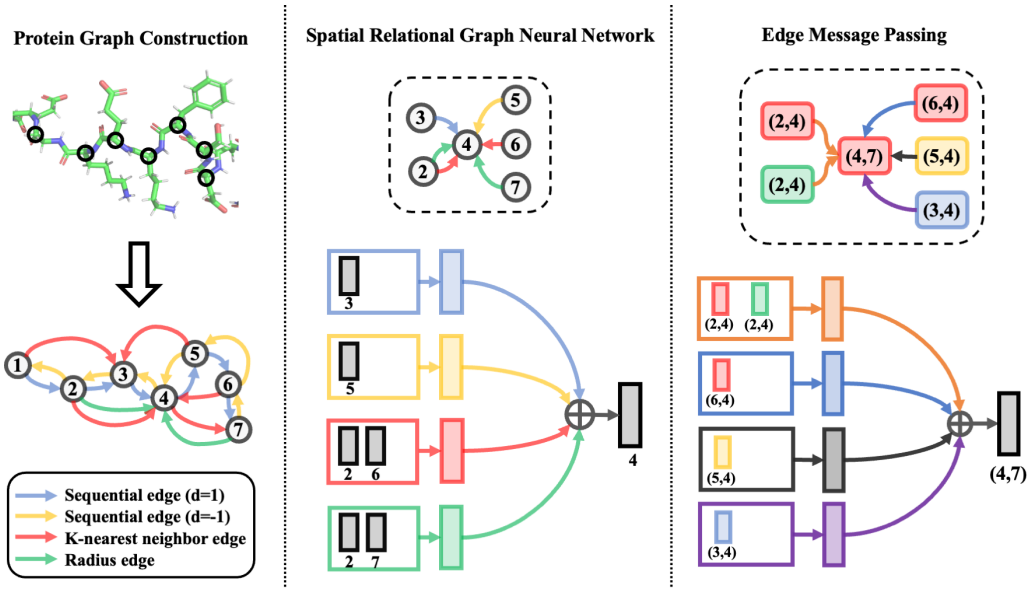

Figure 2: The pipeline for GearNet and GearNet-edge. First, we construct a relational protein residue graph with sequential, radius and knn edges (some edges are omitted in the figure to save space). Then, a relational graph convolutional layer is applied. Similar message passing layers can be applied on the edge graph to improve the model capacity. This figure shows the update iteration for node 4 and edge $(4, 7, \text{red})$, respectively.

to learn good representation as shown in our experiments.

The feature $\boldsymbol{f}_{(i,j,r)}$ for an edge (i, j, r) is the concatenation of the node features of two end nodes, the one-hot encoding of the edge type, and the sequential and spatial distances between them:

$$\boldsymbol{f}_{(i,j,r)} = \text{Cat}\left(\boldsymbol{f}_i, \boldsymbol{f}_j, \text{onehot}(r), |i-j|, \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2\right), \tag{2}$$

where $\text{Cat}(\cdot)$ denotes the concatenation operation.

C.3. Enhance GearNet with IEConv Layers

In our experiments, we find that IEConv layers are very useful for predicting fold labels in spite of their relatively poor performance on function prediction tasks. Therefore, we enhance our models by adding a simplified IEConv layer as an additional layer, which achieve better results than the original IEConv. In this section, we describe how to simplify the IEConv layer and how to combine it with our model.

Simplify the IEConv layer. The original IEConv layer relies on the computation of intrinsic and extrinsic distances between two nodes, which are computationally expensive. Hence, we follow the modifications proposed in (Hermosilla & Ropinski, 2022), which show improvements as reported in their experiments. Although these modifications are not proposed by us, we still briefly describe the model for completeness.

In the IEConv layer, we keep the edges in our graph $\mathcal{G}$ and use $\tilde{\boldsymbol{h}}_i^{(l)}$ to denote the hidden representation for node i in the l-th layer. The update equation for node i is defined as:

$$\tilde{\boldsymbol{h}}_i^{(l)} = \sum_{j \in \mathcal{N}(i)} k_o(f(\mathcal{G}, i, j)) \cdot \boldsymbol{h}_j^{(l-1)}, \tag{3}$$

where $\mathcal{N}(i)$ is the set of neighbors of i, $f(\mathcal{G}, i, j)$ is the edge feature between i and j and $k_o(\cdot)$ is an MLP mapping the feature to a kernel matrix. Instead of intrinsic and extrinsic distances in the original IEConv layer, we follow New IEConv, which adopts three relative positional features proposed in (Ingraham et al., 2019) and further augments them with additional input functions.

We aim to apply this layer on our constructed protein residue graph instead of the radius graph in the original paper. Therefore, we simply remove the dynamically changed receptive fields, pooling layer and smoothing tricks in our setting.

Combine IEConv with GearNet. Our model is very flexible to incorporate other message passing layers. To incorporate IEConv layers, we just use our graph and hidden representations as input and replace the update equation with

$$\boldsymbol{h}_i^{(l)} = \boldsymbol{h}_i^{(l-1)} + \tilde{\boldsymbol{h}}_i^{(l)} + \sigma\left(\text{BN}\left(\sum_{r \in \mathcal{R}} \boldsymbol{W}_r \sum_{j \in \mathcal{N}_r(i)} \boldsymbol{h}_j^{(l-1)}\right)\right).$$

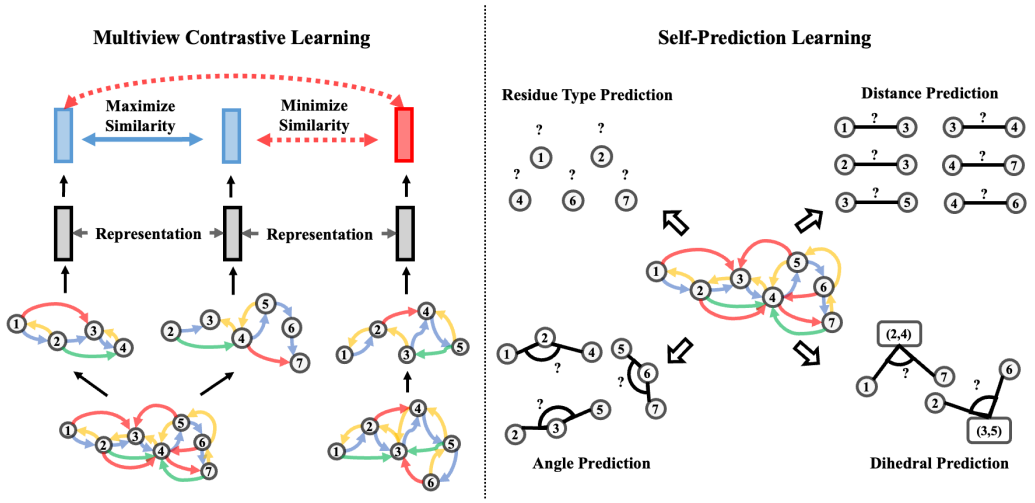

Figure 3: Demonstration of geometric pretraining methods. For multiview contrastive learning, we aim to align representations of different views from the same protein while minimizing the similarity between those from different ones. For self-prediction methods, we construct four masked prediction objectives by inferring masked geometric or biochemical quantities with learned representations.

D. Demonstration of Geometric Preatraining

The high-level ideas of our proposed five geometric pretraining methods are illustrated in Figure 3. For multiview contrastive learning methods, we propose two cropping functions and two transformation functions to generate different views of proteins. Here we illustrate these augmentation functions in Figure 4.

E. Experimental Details

E.1. Setup

Pretraining datasets. We use the AlphaFold protein structure database (CC-BY 4.0 License) (Jumper et al., 2021; Varadi et al., 2021) for pretraining. This database contains the protein structures predicted by the AlphaFold2 model, and we employ both 365K proteome-wide predictions and 440K Swiss-Prot (Consortium, 2021) predictions in our experiments. In Appendix G, we further report the results of pretraining on different datasets.

Downstream tasks. We adopt two tasks proposed in (Gligorijević et al., 2021) and two tasks used in (Hermosilla et al., 2021) for downstream evaluation. ***Enzyme Commission (EC) number prediction*** (BSD 3-Clause License) seeks to predict the EC numbers of different proteins, which describe their catalysis of biochemical reactions. The EC numbers are selected from the third and fourth levels of the EC tree (Webb et al., 1992), forming 538 binary classification tasks. ***Gene Ontology (GO) term prediction*** (BSD 3-Clause License) aims to predict whether a protein belongs to some GO terms. These terms classify proteins into hierarchically related functional classes organized into three ontologies: molecular function (MF), biological process (BP) and cellular component (CC). ***Fold classification*** (CC-BY 4.0 license) is first proposed in (Hou et al., 2018), with the goal to predict the fold class label given a protein. ***Reaction classification*** (BSD 3-Clause License) aims to predict the enzyme-catalyzed reaction class of a protein, in which all four levels of the EC number are employed to depict the reaction class. Although this task is essentially the same with EC prediction, we include it to make a fair comparison with the baselines in (Hermosilla et al., 2021).

Dataset splits. For EC and GO prediction, we follow the multi-cutoff split methods in (Gligorijević et al., 2021) to ensure that the test set only contains PDB chains with sequence identity no more than 95% to the training set, which is also used in (Wang et al., 2022). For fold classification, (Hou et al., 2018) provides three different test sets: *Fold*, in which proteins from the same superfamily are unseen during training; *Superfamily*, in which proteins from the same family are not present during training; and *Family*, in which proteins from the same family are present during training. For reaction classification, we adopt dataset splits proposed in (Hermosilla et al., 2021), where proteins have less than 50% sequence similarity in-between splits.

Baselines. Following (Wang et al., 2022) and (Hermosilla et al., 2021), we compare our encoders with many existing protein representation learning methods, including four sequence-based encoders (CNN (Shanehsazzadeh et al., 2020), ResNet (Rao et al., 2019), LSTM (Rao et al., 2019) and Transformer (Rao et al., 2019)), six

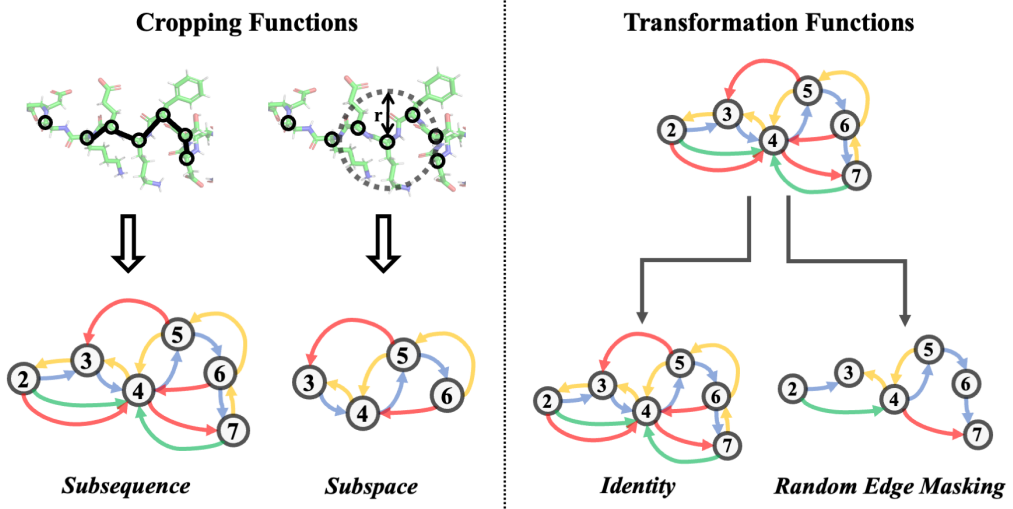

Figure 4: Illustration of four different augmentation functions. First, we randomly apply one of the two cropping functions shown in the figure. For subsequence, we randomly sample a consecutive segment of the protein (2-7 in this case) and take the corresponding subgraph. For subspace, we first sample a center residue (residue 4 in this case) and then sample all residues within a distance threshold r. Then, a random transformation function will be applied on the output subgraph in the last step. For identity, we directly return the graph without transformation, while, for random edge masking, we randomly remove a fixed ratio of edges from the graph.

structure-based encoders (GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), GVP (Jing et al., 2021), 3DCNN_MQA (Derevyanko et al., 2018), GraphQA (Baldassarre et al., 2021) and New IEConv (Hermosilla & Ropinski, 2022)). We also include two models pretrained on large-scale sequence datasets (ProtBERT-BFD (Elnaggar et al., 2021), ESM-1b (Rives et al., 2021)) and two models combining pretrained sequence-based encoders with structural information (DeepFRI (Gligorijević et al., 2021) and LM-GVP (Wang et al., 2022)). For LM-GVP and New IEConv, we only include results reported in the original paper due to the computational burden and the lack of codes.

Training. On the four downstream tasks, we train GearNet and GearNet-Edge from scratch. As we find that the IEConv layer is important for predicting fold labels, we also enhance our model by incorporating this as an additional layer (details in Appendix C.3). These models are referred as GearNet-IEConv and GearNet-IEConv-Edge, respectively. Following previous works (Wang et al., 2022) and (Hermosilla & Ropinski, 2022), the models are trained for 200 epochs on EC and GO prediction and for 300 epochs on fold and reaction classification. For pretraining, the models with the best performance when trained from scratch are selected, *i.e.*, GearNet-Edge for EC, GO, Reaction and GearNet-Edge-IEConv for Fold Classification. The models are pretrained on the AlphaFold protein database with our proposed five methods for 50 epochs. All these models are trained on 4 Tesla A100 GPUs. More details can be found in Appendix E.4.

Evaluation. For EC and GO prediction, we evaluate the performance with the protein-centric maximum F-score F_{max}, which is commonly used in the CAFA challenges (Radivojac et al., 2013) (See Appendix E.3 for details). Experimental results with another popular metric, pair-centric area under precision-recall curve $AUPR_{pair}$, are reported in Appendix. F. For fold and reaction classification, the performance is measured with the mean accuracy. Models with the best performance on validation sets are selected for evaluation on test sets.

E.2. Dataset Statistics

Table 2: Dataset statistics for downstream tasks.

Dataset	# Proteins		
	# Train	# Validation	# Test
Enzyme Commission	15,550	1,729	1,919
Gene Ontology	29,898	3,322	3,415
Fold Classification - *Fold*	12,312	736	718
Fold Classification - *Superfamily*	12,312	736	1,254
Fold Classification - *Family*	12,312	736	1,272
Reaction Classification	29,215	2,562	5,651

Dataset statistics of our four downstream tasks are summarized in Table 2. More details are introduced as follows.

Enzyme Commission and Gene Ontology. Following DeepFRI (Gligorijević et al., 2021), the EC numbers are selected from the third and fourth levels of the EC tree, forming 538 binary classification tasks, while the GO terms with at least 50 and no more than 5000 training samples are selected. The non-redundant sets are partitioned into training, validation and test sets according to the sequence identity. We retrieve all protein chains from PDB used the code provided in their codebase and remove those with obsolete pdb ids, so the statistics will be slightly different from the number reported in the original paper.

Fold Classification. We directly use the dataset processed in (Hermosilla et al., 2021), which consolidated 16,712 proteins with 1,195 different folds from the SCOPe 1.75 database (Murzin et al., 1995).

Reaction Classification. The dataset comprises 37,428 proteins categorized into 384 reaction classes. The split methods are described in (Hermosilla et al., 2021), where they cluster protein chains via sequence similarities and ensure that protein chains belonging to the same cluster are in the same set.

E.3. Evaluation Metrics

Now we introduce the details of evaluation metrics for EC and GO prediction. These two tasks aim to answer the question: whether a protein has a particular function, which can be seen as multiple binary classification tasks.

The first metric, protein-centric maximum F-score $\mathrm{F}_{\max}$, is defined by first calculating the precision and recall for each protein and then taking the average score over all proteins. More specifically, for a given target protein i and some decision threshold $t \in [0, 1]$, the precision and recall are computed as:

$$\mathrm{precision}_i(t) = \frac{\sum_f \mathbb{1}[f \in P_i(t) \cap T_i]}{\sum_f \mathbb{1}[f \in P_i(t)]}, \tag{4}$$

and

$$\mathrm{recall}_i(t) = \frac{\sum_f \mathbb{1}[f \in P_i(t) \cap T_i]}{\sum_f \mathbb{1}[f \in T_i]}, \tag{5}$$

where f is a function term in the ontology, T_i is a set of experimentally determined function terms for protein i, $P_i(t)$ denotes the set of predicted terms for protein i with scores greater than or equal to t and $\mathbb{1}[\cdot] \in \{0, 1\}$ is an indicator function that is equal to 1 *iff* the condition is true.

Then, the average precision and recall over all proteins at threshold t is defined as:

$$\mathrm{precision}(t) = \frac{1}{M(t)} \sum_i \mathrm{precision}_i(t), \tag{6}$$

and

$$\mathrm{recall}(t) = \frac{1}{N} \sum_i \mathrm{recall}_i(t), \tag{7}$$

where we use N to denote the number of proteins and $M(t)$ to denote the number of proteins on which at least one prediction was made above threshold t, *i.e.*, $|P_i(t)| > 0$.

Combining these two measures, the maximum F-score is defined as the maximum value of F-measure over all thresholds. That is,

$$\mathrm{F}_{\max} = \max_t \left\{ \frac{2 \cdot \mathrm{precision}(t) \cdot \mathrm{recall}(t)}{\mathrm{precision}(t) + \mathrm{recall}(t)} \right\}. \tag{8}$$

The second metric, pair-centric area under precision-recall curve $\mathrm{AUPR}_{\mathrm{pair}}$, is defined as the average precision scores for all protein-function pairs, which is exactly the micro average precision score for multiple binary classification.

E.4. Implementation Details

In this subsection, we describe implementation details of all baselines and our methods. For all models, the outputs will be fed into a three-layer MLP to make final prediction. The dimension of hidden layers in the MLP is equal to the dimension of model outputs.

CNN (Shanehsazzadeh et al., 2020). Following the finding in (Shanehsazzadeh et al., 2020), we directly employ a shallow convolutional neural network (CNN) to encode protein sequences. Specifically, 2 convolutional layers with 1024 hidden dimensions and kernel size 5 constitute this baseline model.

ResNet (Rao et al., 2019). We also adopt a deep CNN model, *i.e.*, the ResNet for protein sequences proposed by (Rao et al., 2019), in our benchmark. This model is with 12 residual blocks and 512 hidden dimensions, and it uses the GELU (Hendrycks & Gimpel, 2016) activation function.

LSTM (Rao et al., 2019). The bidirectional LSTM model proposed by (Rao et al., 2019) is another baseline for protein sequence encoding. It is composed of three bidirectional LSTM layers with 640 hidden dimensions.

Transformer (Rao et al., 2019). The self-attention-based Transformer encoder (Vaswani et al., 2017) is a strong model in natural language processing (NLP), (Rao et al., 2019) adapts this model into the field of protein sequence modeling. We also adopt it as one of our baselines. This model has a comparable size with BERT-Small (Devlin et al., 2018), which contains 4 Transformer blocks with 512 hidden dimensions and 8 attention heads, and it is activated by GELU (Hendrycks & Gimpel, 2016).

GCN (Kipf & Welling, 2017). We take GCN as a baseline to encode the residue graph derived by our graph construction scheme. We adopt the implementation in TorchDrug (Zhu et al., 2022), where 6 GCN layers with the hidden dimension of 512 are used for encoding. We run the results of GCN on EC and GO by ourselves and take its results on Fold and Reaction classification from (Hermosilla et al., 2021).

GAT (Veličković et al., 2018). We adopt another popular graph neural network, GAT, as a structure-based baseline model. We follow the implementation in TorchDrug and use 6 GAT layers with the hidden dimension of 512 and 1 attention head per layer for encoding. The results on EC, Fold and Reaction classification are based on our runs, and the results on GO are taken from (Wang et al., 2022).

GVP (Jing et al., 2021). The GVP model (Jing et al., 2021) is a decent protein structure encoder. It iteratively updates the scalar and vector representations of a protein, and these representations possess the merit of invariance and equivariance. In our benchmark, we evaluate this baseline method following the official source code. In specific, 3 GVP layers with 32 feature dimensions (20 scalar and 4 vector channels) constitute the GVP model.

3DCNN_MQA (Derevyanko et al., 2018). We implement the 3DCNN model from the paper (Derevyanko et al., 2018) with a box width of 40.0 and input resolution of $120 \times 120 \times 120$. The model has 6 residual blocks and 128 hidden dimensions with ELU activation function.

GraphQA (Baldassarre et al., 2021). Following the hyperparameters in the original paper, we construct the residue graphs based on bond and spatial information and re-implement the graph neural network in our codebase. The best model has 4 layers with 128 node features, 32 edge features and 512 global features.

New IEConv (Hermosilla & Ropinski, 2022). Since the code for New IEConv has not been made public when the paper is written, we reproduce the method according to the description in the paper and achieve similar results on Fold and Reaction classification tasks. Then, we evaluate the method on EC and GO prediction tasks with the default hyperparameters reported in the original paper and follow the standard training procedure on these two tasks.

DeepFRI (Gligorijević et al., 2021). We also evaluate DeepFRI (Gligorijević et al., 2021) in our benchmark, which is a popular structure-based encoder for protein function prediction. DeepFRI employs an LSTM model to extract residue features and further constructs a residue graph to propagate messages among residues, in which a 3-layer graph convolutional network (GCN) (Kipf & Welling, 2017) is used for message passing. We directly utilize the official model checkpoint for baseline evaluation.

ESM-1b (Rives et al., 2021). Besides the from-scratch sequence encoders above, we also compare with two state-of-the-art pretrained protein language models. ESM-1b (Rives et al., 2021) is a huge Transformer encoder model whose size is larger than BERT-Large (Devlin et al., 2018), and it is pretrained on 24 million protein sequences from UniRef50 (Suzek et al., 2007) by masked language modeling (MLM) (Devlin et al., 2018). In our evaluation, we finetune the ESM-1b model with the learning rate that is one-tenth of that of the MLP prediction head.

ProtBERT-BFD (Elnaggar et al., 2021). The other protein language model evaluated in our benchmark is ProtBERT-BFD (Elnaggar et al., 2021) whose size also excesses BERT-Large (Devlin et al., 2018). This model is pretrained on 2.1 billion protein sequences from BFD (Steinegger & Söding, 2018) by MLM (Devlin et al., 2018). The evaluation of ProtBERT-BFD uses the same learning rate configuration as ESM-1b.

LM-GVP (Wang et al., 2022). To further enhance the effectiveness of GVP (Jing et al., 2021), (Wang et al., 2022) proposed to prepend a protein language model, *i.e.* ProtBERT (Elnaggar et al., 2021), before GVP to additionally utilize protein sequence representations. We also adopt this hybrid model as one of our baselines, and its implementation follows the official source code.

Our methods. For pretraining, we use Adam optimizer with learning rate 0.001 and train a model for 50 epochs. Then, the pretrained model will be finetuned on downstream datasets.

For **Multiview Contrast**, we set the cropping length of subsequence operation as 50, the radius of subspace operation as 15Å, the mask rate of random edge masking operation as 0.15. The temperature τ in the InfoNCE loss function is set as 0.07. When pretraining GearNet-Edge and GearNet-Edge-IEConv, we use 96 and 24 as batch sizes, respectively.

For **Distance Prediction**, we set the number of sampled residue pairs as 256. And the batch size will be set as 128 and 32 for GearNet-Edge and GearNet-Edge-IEConv, respectively. For **Residue Type**, **Angle** and **Dihedral Prediction**, we set the number of sampled residues, residue triplets and residue quadrants as 512. And the batch size will be set as 96 and 32 for GearNet-Edge and GearNet-Edge-IEConv, respectively.

For downstream evaluation, the hidden representations in each layer of GearNet will be concatenated for the final

Table 3: Hyperparameter configurations of our model on different datasets. The batch size reported in the table referrs to the batch size on each GPU. All the hyperparameters are chosen by the performance on the validation set.

Hyperparameter		EC	GO	Fold	Reaction
GNN	#layer	6	6	6	6
	hidden dim.	512	512	512	512
	dropout	0.1	0.1	0.2	0.2
Learning	optimizer	AdamW	AdamW	SGD	SGD
	learning rate	1e-4	1e-4	1e-3	1e-3
	weight decay	0	0	5e-4	5e-4
	batch size	2	2	2	2
	# epoch	200	200	300	300

prediction. Table 3 lists the hyperparameter configurations for different downstream tasks. For the four tasks, we use the same optimizer and number of epochs as in the original papers to make fair comparison. And for EC and GO prediction, we use ReduceLROnPlateau scheduler with factor 0.6 and patience 5, while we use StepLR scheduler with step size 50 and gamma 0.5 for fold and reaction classification.

F. Additional Experimental Results

F.1. AUPR on EC and GO Prediction

In Section 4, we have reported experimental results on EC and GO prediction with F_{max} as the metric. Here we report another popular metric AUPR in Table 4. Note that we still use the best model selected by F_{max} on validation sets. It can be observed that our model can still achieve the best performance on EC prediction in both from scratch and pretrained settings. However, there are still non-trivial gaps between our models with the state-of-the-art results. This probably is because of the inconsistency between the two evaluation metrics. It would be interesting to study the relationship between these two metrics and develop a model good at both. We leave this problem as future works.

F.2. Experiments under Different Sequence Cutoffs

Besides the main experiments, where 95% is used as the sequence identity cutoff for EC and GO dataset splitting, we also test our models and several important baselines under four lower sequence identity cutoffs and show the experimental results in Table 5. It can be observed that, at lower sequence identity cutoffs, our model can still achieve the best performance among models without pretraining and get comparable or better results against ESM-1b after pretraining.

F.3. Ablation Studies

To analyze the contribution of different components in our proposed methods, we perform ablation studies on the EC prediction task. The results are shown in Table 6.

Relational graph convolutional layers. To show the effects of relational convolutional layers, we replace it with graph convolutional layers that share a single kernel matrix among all edges. As reported in the table, results can be significantly improved by using relational convolution, which suggests the importance of treating edges as different types.

Edge message passing layers. We also compare the results of GearNet with and without edge message passing layers, the results of which are shown in Tables 1, 6. It can be observed that the performance consistently increases after performing edge message passing. This demonstrates the effectiveness of our proposed mechanism.

Different augmentations in Multiview Contrast. We investigate the contribution of each augmentation operation proposed in the Multiview Contrast method. Instead of randomly sampling cropping functions and transformations, we pretrain our model with four deterministic combinations of augmentations, respectively. As shown in Table 6, all the four combinations can yield good results, which suggests that arbitrary combinations of the proposed cropping and transformation schemes can yield informative partial views of proteins.

Sampling schemes in Self-Prediction methods. Different sampling schemes may lead to different results for self-prediction methods. We study the effects of sampling schemes using Dihedral Prediction as an example. Instead of sampling dihedral angles formed by three consecutive edges, we try to predict the dihedrals formed by four randomly sampled nodes. We observe that this change of sampling schemes will make the self-prediction task more difficult to solve, which even brings negative effects after pretraining.

G. Pretraining on Different Datasets

We use the AlphaFold protein structure database as our pretraining database, as it contains the largest number of protein structures and is planned to cover over 100 million proteins in the future. However, the structures in this database are not experimentally determined but predicted by AlphaFold2. Therefore, it is interesting to see the performance of our methods when pretraining on different datasets.

To study the effects of the choice of pretraining dataset, we build another dataset using structures extracted from Protein Data Bank (PDB) (Berman et al., 2000). Specifically, we extract 123,505 experimentally-determined protein struc-

Table 4: AUPR on EC and GO prediction. [†] denotes results taken from (Wang et al., 2022). For pretraining, we select the model with the best performance when training from scratch, *i.e.*, GearNet-Edge. We omit the model name and use the pretraining methods to name our pretrained models.

	Method	Pretraining Dataset (Size)	EC	GO		
				BP	MF	CC
w/o pretraining	CNN (Shanehsazzadeh et al., 2020)	-	0.526	0.159	0.351	0.204
	ResNet (Rao et al., 2019)	-	0.590	0.205	0.434	0.214
	LSTM (Rao et al., 2019)	-	0.414	0.156	0.334	0.192
	Transformer (Rao et al., 2019)	-	0.218	0.156	0.177	0.210
	GCN (Kipf & Welling, 2017)	-	0.319	0.136	0.147	0.175
	GAT (Veličković et al., 2018)	-	0.320	0.171†	0.329†	0.249†
	GVP (Jing et al., 2021)	-	0.482	0.224†	0.458†	0.279†
	3DCNN_MQA (Derevyanko et al., 2018)	-	0.029	0.132	0.075	0.144
	GraphQA (Baldassarre et al., 2021)	-	0.543	0.199	0.347	0.265
	New IEConv (Hermosilla & Ropinski, 2022)	-	0.775	**0.273**	**0.572**	**0.316**
	GearNet	-	0.751	0.211	0.490	0.276
	GearNet-IEConv	-	0.835	0.231	0.547	0.259
	GearNet-Edge	-	**0.872**	0.251	**0.570**	0.303
	GearNet-Edge-IEConv	-	0.843	0.244	0.561	0.284
w/ pretraining	DeepFRI (Gligorijević et al., 2021)	Pfam (10M)	0.547	0.282	0.462	0.363
	ESM-1b (Rives et al., 2021)	UniRef50 (24M)	0.889	**0.343**	**0.639**	0.384
	ProtBERT-BFD (Elnaggar et al., 2021)	BFD (2.1B)	0.859	0.188†	0.464†	0.234†
	LM-GVP (Wang et al., 2022)	UniRef100 (216M)	0.710	0.302†	0.580†	**0.423**†
	Multiview Contrast	AlphaFoldDB (805K)	**0.892**	0.292	0.596	0.336
	Residue Type Prediction	AlphaFoldDB (805K)	0.870	0.267	0.583	0.311
	Distance Prediction	AlphaFoldDB (805K)	0.863	0.274	0.586	0.327
	Angle Prediction	AlphaFoldDB (805K)	0.880	0.291	0.603	0.331
	Dihedral Prediction	AlphaFoldDB (805K)	0.881	0.304	0.603	0.338

tures from PDB whose resolutions are between 0.0 and 2.5 angstroms, and we further extract 305,265 chains from these proteins to construct the final pretraining dataset.

Next, we pretrain our five methods on AlphaFold Database v1 (proteome-wide structure predictions), AlphaFold Database v2 (Swiss-Prot structure predictions) and Protein Data Bank and then evaluate the pretrained models on the EC prediction task. The results are reported in Table 7. As can be seen in the table, our methods can achieve comparable performance on different pretraining datasets. Consequently, our methods are robust to the choice of pretraining datasets.

H. Results on Protein Engineering Tasks

Besides four standard tasks considered in Section 4, another important kind of downstream tasks is related to protein engineering, which is heavily relied on mutations on protein sequences. These tasks aim to predict the ability of a protein to perform a desired function, termed protein fitness. Good models are expected to have sufficient precision to distinguish between closely-related protein sequences upon mutations. In this section, we further evaluate our model on four protein engineering related tasks.

H.1. Setup

We choose two protein engineering tasks from (Rao et al., 2019) and two landscape prediction tasks from (Dallago et al., 2021), all of which are standard benchmarks to evaluate protein language models. The statistics of four datasets are shown in Table 8 and we describe each task as follows.

Fluoresence landscape prediction (Sarkisyan et al., 2016) This task aims to predict the log-fluorescence intensity of mutants of the wild type green fluorescent protein (GFP). The training set consists of single, double and triple mutants, while the test set includes variants with four or more mutations.

Stability landscape prediction (Rocklin et al., 2017) This is a regression task that maps each input protein to a value measuring the most extreme circumstances in which the protein can maintain its fold above a concentration threshold. To test the generalization ability from a broad set of relevant sequences with multiple mutations to local sequence neighbors with fewer mutations, the training set includes proteins from four rounds of experimental design, whereas the test set only contains 1-hop neighbors of top candidate proteins.

Table 5: F_{max} on EC and GO tasks under different sequence cutoffs (30% / 40% / 50% / 70% / 95%).

Method	EC	GO-BP	GO-MF	GO-CC
CNN	0.366 / 0.361 / 0.372 / 0.429 / 0.545	0.197 / 0.195 / 0.197 / 0.211 / 0.244	0.238 / 0.243 / 0.256 / 0.292 / 0.354	0.258 / 0.257 / 0.260 / 0.263 / 0.387
ResNet	0.409 / 0.412 / 0.450 / 0.526 / 0.605	0.230 / 0.230 / 0.234 / 0.249 / 0.280	0.282 / 0.288 / 0.308 / 0.347 / 0.405	0.277 / 0.273 / 0.280 / 0.278 / 0.304
LSTM	0.247 / 0.249 / 0.270 / 0.333 / 0.425	0.194 / 0.192 / 0.195 / 0.205 / 0.225	0.223 / 0.229 / 0.245 / 0.276 / 0.321	0.263 / 0.264 / 0.269 / 0.270 / 0.283
Transformer	0.167 / 0.173 / 0.175 / 0.197 / 0.238	0.267 / 0.265 / 0.262 / 0.262 / 0.264	0.184 / 0.187 / 0.195 / 0.204 / 0.211	0.378 / 0.382 / 0.388 / 0.395 / 0.405
GCN	0.245 / 0.246 / 0.246 / 0.280 / 0.320	0.251 / 0.250 / 0.248 / 0.248 / 0.252	0.180 / 0.183 / 0.187 / 0.194 / 0.195	0.318 / 0.318 / 0.320 / 0.323 / 0.329
GearNet	0.557 / 0.570 / 0.615 / 0.693 / 0.730	0.309 / 0.309 / 0.315 / 0.336 / 0.356	0.382 / 0.397 / 0.425 / 0.474 / 0.503	0.381 / 0.385 / 0.393 / 0.398 / 0.414
GearNet-edge	**0.625 / 0.646 / 0.694 / 0.757 / 0.810**	**0.345 / 0.347 / 0.354 / 0.378 / 0.403**	**0.444 / 0.461 / 0.490 / 0.537 / 0.580**	**0.394 / 0.394 / 0.401 / 0.408 / 0.450**
DeepFRI	0.470 / 0.505 / 0.545 / 0.600 / 0.631	0.361 / 0.362 / 0.371 / 0.391 / 0.399	0.374 / 0.383 / 0.409 / 0.446 / 0.465	0.440 / 0.441 / 0.444 / 0.451 / 0.460
ESM-1b	0.737 / 0.764 / 0.797 / 0.839 / 0.864	0.394 / 0.399 / 0.407 / 0.429 / 0.470	**0.546 / 0.562 / 0.588 / 0.625 / 0.657**	**0.462 / 0.465 / 0.468 / 0.465 / 0.488**
Multiview Contrast	**0.744 / 0.769 / 0.808 / 0.848 / 0.874**	**0.436 / 0.442 / 0.449 / 0.471 / 0.490**	0.533 / 0.548 / 0.573 / 0.612 / **0.654**	0.459 / 0.460 / **0.467 / 0.469 / 0.488**

Method	F_{max}
GearNet-Edge	**0.810**
- w/o relational convolution	0.752
- w/o edge message passing	0.730
GearNet-Edge (Multiview Contrast)	**0.874**
- subsequence + identity	0.866
- subspace + identity	**0.872**
- subsequence + random edge masking	0.869
- subspace + random edge masking	**0.876**
GearNet-Edge (Dihedral Prediction)	**0.859**
- w/ random sampling	0.821

Table 6: Ablation studies on EC.

GB1 (Wu et al., 2016) This task uses the GB1 landscape to test the model's ability to predict the effects of interactions between mutations, termed epistasis. We adopt the low-vs-high split proposed in (Dallago et al., 2021), where sequences with fitness value equal to or below wild type are used to train, while sequences with fitness value above wild type are used to test.

Thermostability (Jarzab et al., 2020) We use the screening landscape curated from (Dallago et al., 2021) to measure the model's ability to predict thermostability of proteins. This landscape includes both global and local variation instead of only mutants of a single protein. Similarly, we use the low-vs-high split proposed in (Dallago et al., 2021).

All these four tasks are evaluated via Spearman's ρ (rank correlation coefficient).

H.2. Implementation Details

Protein structure generation. Since all these tasks are originally designed for evaluating sequence-based encoders, the experimentally-determined structures for proteins in these datasets are not available. To solve this issue, we use AlphaFold2 to generate structures for all datasets except GB1. For GB1 dataset, we only generate the structure of the wild type protein. Because the differences between mutant structures are almost negligible on the residue level, we directly use the residue graph constructed from the wild type protein for all mutants and replace node and edge features with corresponding residue types after mutations. We scale up the number of predictions that can be performed by AlphaFold2 by using the fast homology search of MMSeqs2 (Steinegger & Söding, 2017) and running AlphaFold2 in batch mode using ColabFold (Mirdita et al., 2021). Five predictions were made for each sequence and prediction with the highest pLDDT score was chosen. The number of recycles were set to 3 and relaxation using amber force fields was not used. For those proteins the structures of which AlphaFold2 fails to generate, we only add sequential edges in the protein residue graph based on sequential information.

Training details. We use the same pretraining and downstream setting as in Section E.4. Hyperparameters for each dataset are described in Table 9.

H.3. Results

The results on four protein engineering tasks are reported in Table 10. First, our model achieves the best result on all tasks among models without pretraining. Moreover, it obtains comparable or even better performance than pretrained sequence encoders on Fluoresence and Stability tasks. This can be understood because our model is a generalization of CNN enhanced with structural information encoded in the protein graph and thus at least as good as CNN. Surprisingly, the improvements of edge message passing aren't significant on these four tasks. One potential reason is that the structures of protein mutants are undistinguishable on the residue level and some structures are missing in the dataset.

As for pretraining, structure-based methods show significant improvements on Stability and GB1 datasets, which achieve the state-of-the-art performance. However, no positive effects of pretraining are shown on the other two tasks. Besides the reason mentioned above, it may also be attributed to the fact that sequential information plays a more important role in these tasks.

Table 7: Results of GearNet-Edge pretrained on different pretraining datasets with different methods. Models are evaluated on the EC prediction task.

Dataset	# Proteins	Multiview Contrast		Residue Type Prediction		Distance Prediction		Angle Prediction		Dihedral Prediction	
		$AUPR_{pair}$	F_{max}	$AUPR_{pair}$	F_{max}	$AUPR_{pair}$	F_{max}	$AUPR_{pair}$	F_{max}	$AUPR_{pair}$	F_{max}
AlphaFold Database (v1 + v2)	804,872	0.892	0.874	0.870	0.834	0.863	0.839	0.880	0.853	0.881	0.859
AlphaFold Database (v1)	365,198	0.890	0.874	0.869	0.842	0.871	0.843	0.879	0.854	0.877	0.852
AlphaFold Database (v2)	439,674	0.890	0.874	0.868	0.838	0.868	0.846	0.881	0.853	0.883	0.861
Protein Data Bank	305,265	0.881	0.859	0.870	0.841	0.865	0.847	0.880	0.857	0.886	0.858

Table 8: Dataset statistics for protein engineering tasks.

Dataset	# Proteins		
	# Train	# Validation	# Test
Fluorescence	21,446	5,362	27,217
Stability	53,614	2,512	12,851
GB1	4,580	509	3,644
Thermostability	5,149	643	1,366

Table 9: Hyperparameter configurations of our model on protein engineering datasets. The batch size reported in the table referrs to the batch size on each GPU. All the hyperparameters are chosen by the performance on the validation set.

Hyperparameter		Fluores	Stability	GB1	Thermo
GNN	#layer	6	6	6	6
	hidden dim.	512	512	512	512
	dropout	0.1	0.1	0.1	0.1
Learning	optimizer	AdamW	AdamW	AdamW	AdamW
	learning rate	1e-4	1e-4	1e-4	1e-4
	weight decay	0	0	0	0
	batch size	8	32	8	2
	# epoch	200	200	200	200

Overall, our methods achieve the state-of-the-art performance on three of four datasets. Nevertheless, it still needs further exploration and ablation studies on these mutation-based datasets, which should be considered in future work.

I. Latent Space Visualization

For qualitatively evaluating the quality of the protein embeddings learned by our pretraining method, we visualize the latent space of the GearNet-Edge model pretrained by Multiview Contrast. Specifically, we utilize the pretrained model to extract the embeddings of all the proteins in AlphaFold Database v1, and these embeddings are mapped to the two-dimensional space by UMAP (McInnes et al., 2018) for visualization. Following (Akdel et al., 2021), we highlight the 20 most common superfamilies within the database by different colors. The visualization results are shown in Fig. 5. It can be observed that our pretrained model tends to group the proteins from the same superfamily together and divide the ones from different superfamilies apart. In particular, it succeeds in clearly separating three superfamilies, *i.e.*, Protein kinase superfamily, Cytochrome P450 family and TRAFAC class myosin-kinesin ATPase superfamily. Such a decent capability of discriminating protein superfamilies, to some degree, interprets our model's superior performance on Fold Classification.

J. Residue-Level Explanation

Protein functions are often reflected by specific regions on the 3D protein structures. For example, the binding ability of a protein to a ligand is highly related to the binding interface between them. Hence, to better interpret our prediction, we apply Integrated Gradients (IG) (Sundararajan et al., 2017), a model-agnostic attribution method, on our model to obtain residue-level interpretation. Specifically, we first select two molecular functions, ATP binding (GO:0005524) and Heme binding (GO:0020037), from GO terms that are related to ligand binding. For each functional term, we pick one protein and feed it into the best model trained on the GO-MF dataset. Then, we use IG to generate the feature attribution scores for each protein. The method will integrate the gradient along a straight-line path between a baseline input and the original input. Here the original input and baseline input are the node feature $\boldsymbol{f}$ and a zero vector, respectively. The final attribution score for each protein will be obtained by summing over the feature dimension. The normalized score distribution over all residues are visualized in Figure 6. As can be seen, our model is able to identify the active sites around the ligand, which are likely to be responsible for binding. Note that these attributions are directly generated from our model without any supervision, which suggests the decent interpretability of our model.

Table 10: Spearman's ρ (rank correlation coefficient) on four protein engineering tasks. [*] means the results are taken from (Wang et al., 2022), while [†] means the results are taken from (Dallago et al., 2021).

Category	Method	Fluores	Stability	GB1	Thermo
	Without Pretraining				
Sequence-based	CNN (Shanehsazzadeh et al., 2020)	0.656	0.717	0.51†	0.49†
	ResNet (Rao et al., 2019)	0.369	0.478	0.294	0.412
	LSTM (Rao et al., 2019)	0.124	0.477	0.552	0.142
	Transformer (Rao et al., 2019)	0.522	0.645	0.001	OOM
Structure-based	GAT (Veličković et al., 2018)	0.390*	0.565*	-	-
	GVP (Jing et al., 2021)	0.545*	0.680*	-	-
	New IEConv (Hermosilla & Ropinski, 2022)	0.635	0.529	0.205	OOM
Ours	**GearNet**	**0.682**	0.719	**0.546**	0.632
	GearNet-Edge	0.677	**0.740**	**0.545**	**0.654**
	With Pretraining				
Sequence Pretrained	ESM-1b (Rives et al., 2021)	**0.682**	0.734	**0.59**†	**0.76**†
	ProtBERT-BFD (Elnaggar et al., 2021)	0.677*	0.734*	-	-
	LM-GVP (Wang et al., 2022)	0.679*	0.733*	-	-
Ours	**Multiview Contrast**	0.675	0.752	0.296	0.640
	Residue Type Prediction	0.675	**0.787**	**0.597**	0.543
	Distance Prediction	0.675	0.688	0.583	0.622
	Angle Prediction	0.673	0.727	0.512	0.645
	Dihedral Prediction	0.670	0.746	0.521	0.634

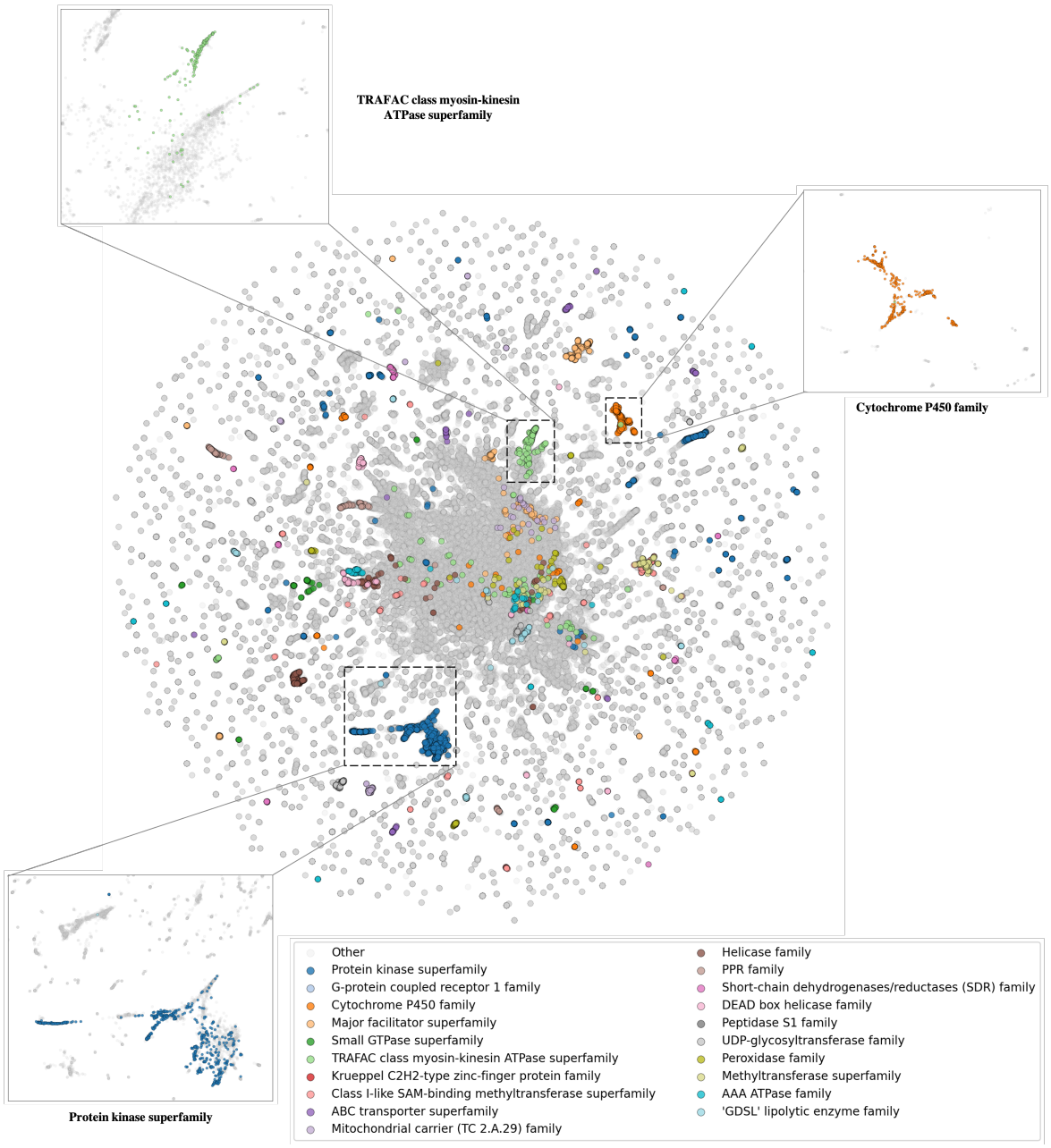

Figure 5: Latent space visualization of GearNet-Edge (Multiview Contrast) on AlphaFold Database v1.

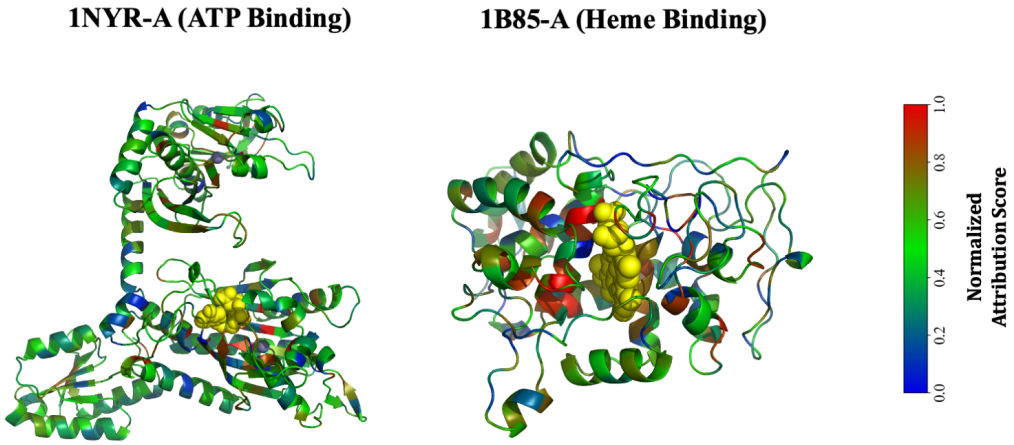

Figure 6: Identification of active sites on proteins responsible for binding based on attribution scores. Two proteins binding to specific targets are selected for illustration (1NYR-A for ATP binding and 1B85-A for Heme binding). For these two complexes, ligands are shown in yellow spheres while the residues of the receptors are colored based on attribution scores. Residues with higher attribution scores are colored in red while those with lower scores are colored in blue.