Geometry-Complete Perceptron Networks for 3D Molecular Graphs

Alex Morehead, Jianlin Cheng
University of Missouri-Columbia
acmwhb@umsystem.edu, chengji@umsystem.edu

Abstract

The field of geometric deep learning has had a profound impact on the development of innovative and powerful graph neural network architectures. Disciplines such as computer vision and computational biology have benefited significantly from such methodological advances, which has led to breakthroughs in scientific domains such as protein structure prediction and design. In this work, we introduce GCPNET, a new geometry-complete, SE(3)-equivariant graph neural network designed for 3D graph representation learning. We demonstrate the state-of-the-art utility and expressiveness of our method on six independent datasets designed for three distinct geometric tasks: protein-ligand binding affinity prediction, protein structure ranking, and Newtonian many-body systems modeling. Our results suggest that GCPNET is a powerful, general method for capturing complex geometric and physical interactions within 3D graphs for downstream prediction tasks. Our code and data are available at https://github.com/BioinfoMachineLearning/GCPNet.

1 Introduction

Begin a ubiquitous form of information, graph-structured data arises from numerous sources such as the fields of physics and chemistry. Moreover, the relational nature of graph-structured data allows one to identify and characterize topological associations between entities in large real-world networks (e.g., social networks). In particular, 3D data often emerges in domains such as computer vision and can be readily described as graph-structured inputs. To process and analyze such 3D information in a meaningful and powerful way, one must carefully consider the symmetries present in such data to reduce the geometric redundancies they might present to a machine learning model.

1.1 Related Work

Previous works in geometric deep learning (Bronstein et al. 2021) have explored the use of neural networks for modeling physical systems (Cao et al. 2020). Some of the earliest neural networks applied to physical systems include convolutional networks (CNNs) (LeCun, Bengio et al. 1995), graph neural networks (GNNs) (Kipf and Welling 2016), and point cloud neural networks (Qi et al. 2017). Throughout their development, such geometric deep learning methods have expanded to incorporate within their individual layers equivariance to various geometric symmetry groups to enhance their generalization capabilities and adversarial robustness. Methods such as group-equivariant CNNs (Cohen and Welling 2016), Tensor Field Networks (Thomas et al. 2018), SE(3)-Transformers (Fuchs et al. 2020), and equivariant GNNs (Fuchs et al. 2020; Jing et al. 2020, 2021; Du et al. 2022; Aykent and Xia 2022) have paved the way for the development of future deep learning models that respect physical symmetries present in 3D data (e.g., equivariance with respect to rotation symmetries occurring in input data).

1.2 Contributions

In this work, we make connections between geometric graph neural networks, equivariance, and geometry information completeness guarantees that provide one with a rich foundation on which to build new graph neural network architectures. In particular, we introduce a new graph neural network model that is equivariant to the group of 3D rotations and translations (i.e., the SE(3) group) and guarantees directional information completeness following graph message-passing on 3D point clouds. We showcase its expressiveness and flexibility for modeling physical systems through several benchmark studies. In detail, we provide the following contributions.

- We present the first geometric graph neural network architecture with directional information completeness guarantees that in an SE(3)-equivariant manner can predict new node positions as well as scalar and vector-valued features for nodes and edges.
- We establish new state-of-the-art results for three separate molecular-geometric representation learning tasks where model predictions vary from analyzing individual nodes to summarizing entire graph inputs.
- Our experiments demonstrate that the geometric information that rich geometric message-passing procedures and local equivariant frame encodings of node positions provide is useful for predicting both vector-valued node features as well as scalar node and graph-level properties across different geometric datasets.

2 Proposed Architecture

2.1 Overview of Our Approach

We represent a 3D molecular structure as a 3D k-nearest neighbors (k-NN) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathbf{X} \in \mathbb{R}^{N \times 3}$ as the respective Cartesian coordinates for each node, where $N = |\mathcal{V}|$ and $E = |\mathcal{E}|$. We then design E(3)-invariant (i.e., 3D rotation, reflection, and translation-equivariant) node features $\mathbf{H} \in \mathbb{R}^{N \times h}$ and edge features $\mathbf{E} \in \mathbb{R}^{E \times e}$ as well as O(3)-equivariant (3D rotation and reflection-equivariant) node features $\boldsymbol{\chi} \in \mathbb{R}^{N \times (m \times 3)}$ and edge features $\boldsymbol{\xi} \in \mathbb{R}^{E \times (x \times 3)}$, respectively.

Upon constructing such features, we apply several layers of graph message-passing using functions Φ that update node and edge features using invariant and equivariant representations for the corresponding feature types. Importantly, our method for doing so guarantees, by design, *SE(3) equivariance* with respect to its vector-valued input coordinates and features (i.e., $x_i \in \mathbf{X}$, $\chi_i \in \boldsymbol{\chi}$, and $\xi_{ij} \in \boldsymbol{\xi}$) and *SE(3)-invariance* regarding its scalar features (i.e., $h_i \in \mathbf{H}$ and $e_{ij} \in \mathbf{E}$) to achieve *geometric self-consistency* of the 3D structure of the input molecular graph $\mathcal{G}$ during graph message-passing. We formalize the equivariance, geometric self-consistency, and geometric completeness constraints using the following three definitions.

Definition 1 (SE(3) Equivariance)**.**

$$\begin{aligned}
&\text{Given } (\mathbf{H}', \mathbf{E}', \mathbf{X}', \boldsymbol{\chi}', \boldsymbol{\xi}') = \boldsymbol{\Phi}(\mathbf{H}, \mathbf{E}, \mathbf{X}, \boldsymbol{\chi}, \boldsymbol{\xi}) \\
&\text{we have } (\mathbf{H}', \mathbf{E}', \mathbf{Q}\mathbf{X}'^{\mathrm{T}} + \mathbf{g}, \mathbf{Q}\boldsymbol{\chi}'^{\mathrm{T}}, \mathbf{Q}\boldsymbol{\xi}'^{\mathrm{T}}) = \\
&\boldsymbol{\Phi}(\mathbf{H}, \mathbf{E}, \mathbf{Q}\mathbf{X}^{\mathrm{T}} + \mathbf{g}, \mathbf{Q}\boldsymbol{\chi}^{\mathrm{T}}, \mathbf{Q}\boldsymbol{\xi}^{\mathrm{T}}), \\
&\quad \forall \mathbf{Q} \in SO(3), \forall \mathbf{g} \in \mathbb{R}^{3 \times 1}.
\end{aligned} \tag{1}$$

Definition 2 (Geometric Self-Consistency)**.**

$$\begin{aligned}
&\text{Given a pair of molecular graphs } \mathcal{G}_1 \text{and } \mathcal{G}_2, \\
&\text{with } \mathbf{X}^1 = \{\mathbf{x}_i^1\}_{i=1,\dots,N} \text{ and } \mathbf{X}^2 = \{\mathbf{x}_i^2\}_{i=1,\dots,N}, \\
&\text{respectively, a geometric representation} \\
&\boldsymbol{\Phi}(\mathbf{H}, \mathbf{E}) = \boldsymbol{\Phi}(\mathcal{G}) \text{ is considered geometrically complete} \\
&\text{if } \boldsymbol{\Phi}(\mathcal{G}^1) = \boldsymbol{\Phi}(\mathcal{G}^2) \iff \exists \mathbf{Q} \in SO(3), \exists \mathbf{g} \in \mathbb{R}^{3 \times 1}, \\
&\text{for } i = 1, ..., n, \mathbf{X}_i^{1^{\mathrm{T}}} = \mathbf{Q}\mathbf{X}_i^{2^{\mathrm{T}}} + \mathbf{g}.
\end{aligned} \tag{2}$$

Definition 3 (Geometric Completeness)**.**

$$\begin{aligned}
&\text{Given a positional pair of nodes } (x_i^t, x_j^t) \text{ in a 3D graph } \mathcal{G}, \\
&\text{with vectors } a_{ij}^t \in \mathbb{R}^{1 \times 3},\ b_{ij}^t \in \mathbb{R}^{1 \times 3}, \text{ and } c_{ij}^t \in \mathbb{R}^{1 \times 3} \\
&\text{derived from } (x_i^t, x_j^t), \text{ a local geometric representation} \\
&\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t) \in \mathbb{R}^{3 \times 3} \text{ is considered geometrically} \\
&\text{complete if } \boldsymbol{\mathcal{F}}_{ij}^t \text{ is non-degenerate, thereby forming a} \\
&\textit{local orthonormal basis} \text{ located at the tangent space of } x_i^t.
\end{aligned} \tag{3}$$

2.2 SE(3)-equivariant complete representations

Representation learning on 3D molecular structures is a challenging task for a variety of reasons: (1) an expressive representation learning model should be able to predict arbitrary vector-valued quantities for each atom and atom pair in the molecular structure (e.g., using $\boldsymbol{\chi}'$ and $\boldsymbol{\xi}'$ to predict side-chain atom positions and atom-atom displacements for each residue in a 3D protein graph); (2) arbitrary rotations or translations to a 3D molecular structure should affect only the vector-valued representations a model assigns to a molecular graph's nodes or edges, whereas such 3D transformations of the molecular structure should not affect the model's scalar representations for nodes and edges (Du et al. 2022); (3) the geometrically invariant properties of a molecule's 3D structure should be uniquely identifiable by a model; and (4) in a geometry-complete manner, scalar and vector-valued representations should mutually exchange information between nodes and edges during a model's forward pass for a 3D input graph, as these information types can be correlatively related (e.g., a scalar feature such as the L_2 norm of a vector v can be associated with the vector of origin v) (Aykent and Xia 2022; Morehead, Chen, and Cheng 2022).

In line with this reasoning, we need to ensure that the coordinates our model predicts for the node positions in a molecular graph $\mathcal{G}$ transform according to SE(3) transformations of the input positions, in contrast to our current approaches that remain E(3)-equivariant or E(3)-invariant to 3D positional transformations of $\mathcal{G}$ and consequently introduce insufficient geometric priors into the model's learning procedure (e.g., due to the chirality of 3D protein structures). Simultaneously, without introducing any *direction degeneration* between pairs of node positions, the model should SE(3)-invariantly and SE(3)-equivariantly update the scalar and vector-valued features of $\mathcal{G}$, respectively. To increase its generalization capabilities, our model should also maintain SE(3)-invariance of its scalar features produced when the input graph is transformed in 3D space. Following (Wang et al. 2022), this helps prevent the model from losing important geometric information (i.e., attaining geometric self-consistency) during graph message-passing. One way to do this is to introduce a new type of message-passing neural network.

2.3 GCPNet Architecture

Towards this end, we introduce our architecture for $\boldsymbol{\Phi}$ satisfying Eqs. (1), (2), and (3) which we refer to as the Geometry-Complete SE(3)-Equivariant Perceptron Network (GCPNet). Subsequently, our definition for GCPNet is presented in Algorithm 1. We display the design and operations wherein of our proposed GCP module in Figure 1.

It is then straightforward to prove the following three propositions (see Appendices A.1 through A.6 for a more detailed description of the GCPNet architecture and its equivariant properties).

- **Proposition 1.** GCPNets *are SE(3)-equivariant* $\longrightarrow$ *Eq. (1).*
- **Proposition 2.** GCPNets *are geometry self-consistent* $\longrightarrow$ *Eq. (2).*
- **Proposition 3.** GCPNets *are geometry-complete* $\longrightarrow$ *Eq. (3).*

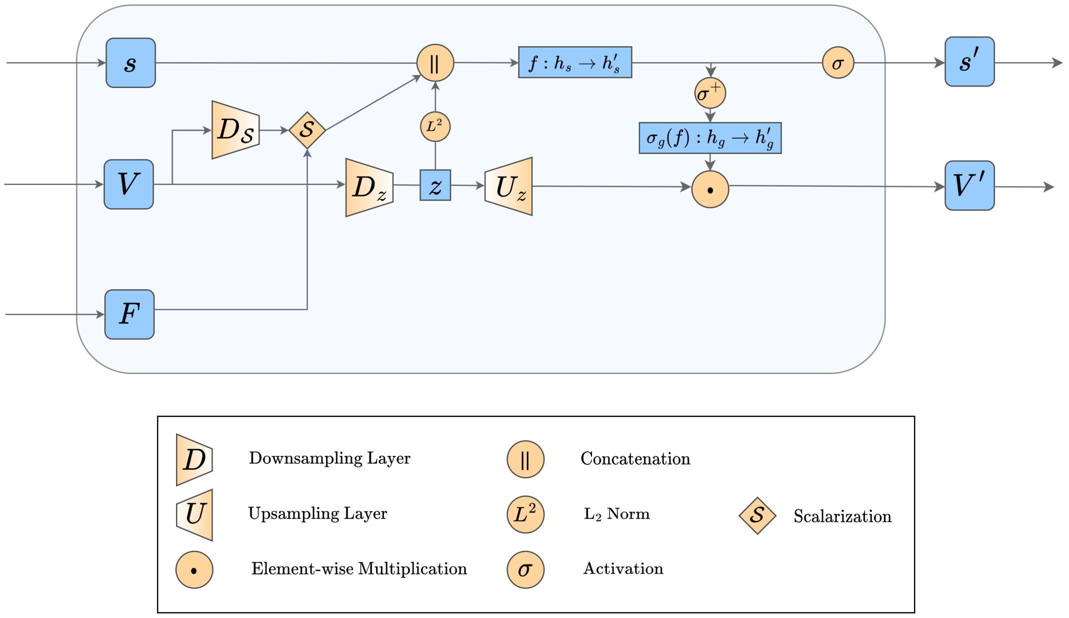

Figure 1: An overview of our proposed Geometry-Complete Perceptron (GCP) module.

3 Experiments

In this work, we consider three distinct modeling tasks, comprised of six datasets in total.

LBA, Graph Regression. Protein-ligand binding affinity prediction (LBA) challenges methods to estimate the binding affinity of a protein-ligand complex as a single scalar value (Townshend et al. 2020). Accurately estimating such values in a matter of seconds using a machine learning model can provide invaluable and timely information in the typical drug discovery pipeline (Rezaei et al. 2020). The corresponding dataset for this SE(3)-invariant task is derived from the ATOM3D dataset (Townshend et al. 2020) and is comprised of 4,463 nonredundant protein-ligand complexes. Results are reported in terms of the root mean squared error (RMSE), Pearson's correlation (p), and Spearman's correlation (Sp) between a method's predictions on the test dataset and the corresponding ground-truth binding affinity values.

PSR, Graph Regression. Protein structure ranking (PSR) requires methods to predict the overall quality of a 3D protein structure when comparing it to a reference (i.e., native) protein structure (Townshend et al. 2020). The quality of a protein structure is reported as a single scalar value representing a method's predicted global distance test (GDT_TS) score (Zemla 2003) between the provided decoy structure and the native structure. Such information is crucial in drug discovery efforts when one is tasked with designing a drug (e.g., ligand) that should bind to a particular protein target, notably when such targets have not yet had their 3D structures experimentally determined and have rather had them predicted computationally using methods such as AlphaFold 2 (Jumper et al. 2021). The respective dataset for this SE(3)-invariant task is also derived from the ATOM3D dataset (Townshend et al. 2020) and is comprised of 40,950 decoy structures corresponding to 649 total targets. Results are reported in terms of the Pearson's correlation (p), Spearman's correlation (Sp), and Kendall's tau correlation (K) between a method's predictions on the test dataset and the corresponding ground-truth GDT_TS values, where local results are averaged across predictions for individual targets and global results are averaged directly across all targets.

NMS, Node Regression. Newtonian many-body systems modeling (NMS) asks methods to forecast the future positions of particles in many-body systems of various sizes (Du et al. 2022), bridging the gap between the domains of machine learning and physics. In our experimental results for the NMS task, the four systems (i.e., datasets) on which we evaluate each method are comprised of increasingly more nodes and are influenced by force fields of increasingly complex directional origins for which to model, namely electrostatic force fields for 5-body (ES(5)) and 20-body (ES(20)) systems as well as for 20-body systems under the influence of an additional gravity field (G+ES(20)) and Lorentz-like force field (L+ES(20)), respectively. The four datasets for this SE(3)-equivariant task were generated using the descriptions and source code of (Du et al. 2022), where each dataset is comprised of 7,000 total trajectories. Results are reported in terms of the mean squared error between a method's node position predictions on the test dataset and the ground-truth node positions after 1,000 timesteps.

4 Analysis and Discussion

The results shown in Table 1 reveal that GCPNET achieves the best performance for predicting protein-ligand binding affinity with a notable margin. The results in Table 2 demonstrate that GCPNET also performs best against all other models for the task of estimating a 3D protein struc-

Algorithm 1: GCPNet

```
Input: $(h_i \in \mathbf{H}, \chi_i \in \boldsymbol{\chi}), (e_{ij} \in \mathbf{E}, \xi_{ij} \in \boldsymbol{\xi}), x_i \in \mathbf{X}$, graph $\mathcal{G}$
1  Initialize $\mathbf{X}^0 = \mathbf{X}^C \leftarrow \mathbf{Centralize}(\mathbf{X})$
2  $\mathcal{F}_{ij} = \mathbf{Localize}(x_i \in \mathbf{X}^0, x_j \in \mathbf{X}^0)$
3  Project $(h_i^0, \chi_i^0), (e_{ij}^0, \xi_{ij}^0) \leftarrow \mathbf{GCPEmbed}((h_i, \chi_i), (e_{ij}, \xi_{ij}), \mathcal{F}_{ij})$
4  for $l \leftarrow 1$ to $L$ by $1$ do
5      $(m_{e_{ij}}^{l-1}, m_{\xi_{ij}}^{l-1}) = \mathbf{Concat}((h_i^{l-1}, \chi_i^{l-1}), (h_j^{l-1}, \chi_j^{l-1}), (e_{ij}^0, \xi_{ij}^0))$
6      $(m_{e_{ij}}^l, m_{\xi_{ij}}^l) = \mathbf{ResGCP}_m^l((m_{e_{ij}}^{l-1}, m_{\xi_{ij}}^{l-1}), \mathcal{F}_{ij})$
7      $(m_{h_i}^l, m_{\chi_i}^l) = \frac{1}{k'} \sum_{j \in \mathcal{N}(i)} (m_{e_{ij}}^l, m_{\xi_{ij}}^l)$
8      $(h_i^l, \chi_i^l) = \mathbf{LayerNorm}((h_i^{l-1}, \chi_i^{l-1}) + \mathbf{Dropout}((m_{h_i}^l, m_{\chi_i}^l)))$
9      $(h_i^l, \chi_i^l) = \mathbf{LayerNorm}((h_i^l, \chi_i^l) + \mathbf{Dropout}(\mathbf{GCP}_{ffn}^l((h_i^l, \chi_i^l))))$
10     if Updating Node Positions then
11         $(h_{v_i}^l, \chi_{v_i}^l) = \mathbf{GCP}_v^l((h_i^l, \chi_i^l), \mathcal{F}_{ij}^l)$
12         $x_i^l = x_i^{l-1} + \chi_{v_i}^l$
13 if Updating Node Positions then
14     $\mathcal{F}_{ij} = \mathbf{Localize}(x_i \in \mathbf{X}^l, x_j \in \mathbf{X}^l)$
15     Finalize $(\mathbf{X}^L) \leftarrow \mathbf{Decentralize}(\mathbf{X}^l)$
16 else
17     $x_i^L = x_i^0$
18 Project $(h_i^L, \chi_i^L), (e_{ij}^L, \xi_{ij}^L) \leftarrow \mathbf{GCPProject}((h_i^l, \chi_i^l), (e_{ij}^0, \xi_{ij}^0), \mathcal{F}_{ij})$
Output: $(h_i^L, \chi_i^L), (e_{ij}^L, \xi_{ij}^L), x_i^L$
```

ture's quality. Lastly, the results shown in Table 3 indicate that GCPNet achieves the best results for two of the four NMS datasets considered in this work, where these two datasets are respectively the first and third most difficult NMS datasets for methods to model, yielding the lowest average mean squared error across all four datasets. Importantly, GCPNet notably improves upon or maintains the performance of all previous methods for both node-level (e.g., NMS) and graph-level (e.g., LBA) prediction tasks, verifying our method's ability to encode useful information for both scales of granularity.

5 Conclusion

In this work, we introduced GCPNet, a state-of-the-art graph neural network for 3D graph representation learning. We have demonstrated its effectiveness through several benchmark studies that suggest that GCPNet is a powerful, general-purpose geometric deep learning method for 3D data. Future work could involve research into developing variations of GCPNet with improved runtime efficiencies or could include additional applications of GCPNet for various other scientific tasks and deep learning datasets.

Table 1: Comparison of GCPNet with baseline methods for the LBA task. The results are averaged over three independent runs.

Type	Method	RMSE $\downarrow$	$p \uparrow$	$Sp \uparrow$
ENN	Cormorant (2019)	1.568 $\pm$ 0.012	0.389	0.408
CNN	3DCNN (2020)	1.416 $\pm$ 0.021	0.550	0.553
	DeepAffinity (2019)	1.893 $\pm$ 0.650	0.415	0.426
Graph	GCN (2020)	1.570 $\pm$ 0.025	0.545	0.533
	DGAT (2021)	1.719 $\pm$ 0.047	0.464	0.472
	DGIN (2021)	1.765 $\pm$ 0.076	0.426	0.432
	DGAT-GCN (2021)	1.550 $\pm$ 0.017	0.498	0.496
	GVP (2021)	1.594 $\pm$ 0.073	0.434	0.432
	GBP (2022)	1.405 $\pm$ 0.009	0.561	0.557
Ours	GCPNet	**1.352 $\pm$ 0.003**	**0.608**	**0.607**

Table 2: Comparison of GCPNet with baseline methods for the PSR task. Local metrics are averaged across target-aggregated metrics.

	Local			Global		
Method	$p \uparrow$	$Sp \uparrow$	$K \uparrow$	$p \uparrow$	$Sp \uparrow$	$K \uparrow$
3DCNN (2020)	0.491	0.431	0.272	0.643	0.769	0.481
ProQ3D (2017)	0.444	0.432	0.304	0.796	0.772	0.594
VoroMQA (2017)	0.412	0.419	0.291	0.688	0.651	0.505
RWplus (2010)	0.192	0.167	0.137	0.033	0.056	0.011
SBROD (2019)	0.431	0.413	0.291	0.551	0.569	0.393
Ornate (2019)	0.393	0.371	0.256	0.625	0.669	0.481
DimeNet (2020)	0.302	0.351	0.285	0.614	0.625	0.431
GraphQA (2021)	0.357	0.379	0.251	0.821	0.820	0.618
GVP (2021)	0.581	0.462	0.331	0.805	0.811	0.616
GBP (2022)	0.612	0.517	0.372	0.856	0.853	0.656
GCPNet	**0.616**	**0.534**	**0.385**	**0.871**	**0.869**	**0.676**

Table 3: Comparison of GCPNet with baseline methods for the NMS task. The final column reports each method's average mean squared error across all four test datasets.

Method	ES(5)	ES(20)	G+ES(20)	L+ES(20)	Average
GNN (2022)	0.0131	0.0720	0.0721	0.0908	0.0620
TFN (2022)	0.0236	0.0794	0.0845	0.1243	0.0780
SE(3)-Transformer (2022)	0.0329	0.1349	0.1000	0.1438	0.1029
Radial Field (2022)	0.0207	0.0377	0.0399	0.0779	0.0441
EGNN (2022)	0.0079	0.0128	0.0118	0.0368	0.0173
ClofNet (2022)	**0.0065**	0.0073	**0.0072**	0.0251	0.0115
GCPNet	0.0070	**0.0071**	0.0073	**0.0173**	**0.0097**

6 Acknowledgments

This work is supported by two NSF grants (DBI1759934 and IIS1763246), two NIH grants (R01GM093123 and R01GM146340), three DOE grants (DE-AR0001213, DE-SC0020400, and DE-SC0021303), and allocations on the Summit cluster provided by Oak Ridge National Laboratory.

References

Anderson, B.; Hy, T. S.; and Kondor, R. 2019. Cormorant: Covariant molecular neural networks. *Advances in neural information processing systems*, 32.

Aykent, S.; and Xia, T. 2022. GBPNet: Universal Geometric Representation Learning on Protein Structures. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, 4–14. New York, NY, USA: Association for Computing Machinery. ISBN 9781450393850.

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

Bronstein, M. M.; Bruna, J.; Cohen, T.; and Veličković, P. 2021. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*.

Cao, W.; Yan, Z.; He, Z.; and He, Z. 2020. A comprehensive survey on geometric deep learning. *IEEE Access*, 8: 35929–35949.

Cohen, T.; and Welling, M. 2016. Group equivariant convolutional networks. In *International conference on machine learning*, 2990–2999. PMLR.

Du, W.; Zhang, H.; Du, Y.; Meng, Q.; Chen, W.; Zheng, N.; Shao, B.; and Liu, T.-Y. 2022. SE(3) Equivariant Graph Neural Networks with Complete Local Frames. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 5583–5608. PMLR.

Falcon, e. a., WA. 2019. PyTorch Lightning. *https://github.com/PyTorchLightning/pytorch-lightning*, 3.

Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. *ArXiv*, abs/1903.02428.

Fuchs, F.; Worrall, D.; Fischer, V.; and Welling, M. 2020. SE(3)-transformers: 3d roto-translation equivariant attention networks. *Advances in Neural Information Processing Systems*, 33: 1970–1981.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Jing, B.; Eismann, S.; Soni, P. N.; and Dror, R. O. 2021. Equivariant graph neural networks for 3d macromolecular structure. *arXiv preprint arXiv:2106.03843*.

Jing, B.; Eismann, S.; Suriana, P.; Townshend, R. J.; and Dror, R. 2020. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*.

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

Karasikov, M.; Pagès, G.; and Grudinin, S. 2019. Smooth orientation-dependent scoring function for coarse-grained protein quality assessment. *Bioinformatics*, 35(16): 2801–2808.

Karimi, M.; Wu, D.; Wang, Z.; and Shen, Y. 2019. DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks. *Bioinformatics*, 35(18): 3329–3338.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Klicpera, J.; Groß, J.; and Günnemann, S. 2020. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*.

LeCun, Y.; Bengio, Y.; et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10): 1995.

Morehead, A.; Chen, C.; and Cheng, J. 2022. Geometric Transformers for Protein Interface Contact Prediction. In *International Conference on Learning Representations*.

Nguyen, T.; Le, H.; Quinn, T. P.; Nguyen, T.; Le, T. D.; and Venkatesh, S. 2021. GraphDTA: Predicting drug–target binding affinity with graph neural networks. *Bioinformatics*, 37(8): 1140–1147.

Olechnovič, K.; and Venclovas, Č. 2017. VoroMQA: Assessment of protein structure quality using interatomic contact areas. *Proteins: Structure, Function, and Bioinformatics*, 85(6): 1131–1145.

Pagès, G.; Charmettant, B.; and Grudinin, S. 2019. Protein model quality assessment using 3D oriented convolutional neural networks. *Bioinformatics*, 35(18): 3313–3319.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Rezaei, M. A.; Li, Y.; Wu, D.; Li, X.; and Li, C. 2020. Deep learning in drug design: protein-ligand binding affinity prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Thomas, N.; Smidt, T. E.; Kearnes, S. M.; Yang, L.; Li, L.; Kohlhoff, K.; and Riley, P. F. 2018. Tensor Field Networks: Rotation- and Translation-Equivariant Neural Networks for 3D Point Clouds. *ArXiv*, abs/1802.08219.

Townshend, R. J.; Vögele, M.; Suriana, P.; Derry, A.; Powers, A.; Laloudakis, Y.; Balachandar, S.; Jing, B.; Anderson, B.; Eismann, S.; et al. 2020. Atom3d: Tasks on molecules in three dimensions. *arXiv preprint arXiv:2012.04035*.

Uziela, K.; Menéndez Hurtado, D.; Shu, N.; Wallner, B.; and Elofsson, A. 2017. ProQ3D: improved model quality assessments using deep learning. *Bioinformatics*, 33(10): 1578–1580.

Van Rossum, G.; and Drake, F. L. 2009. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace. ISBN 1441412697.

Wang, L.; Liu, H.; Liu, Y.; Kurtin, J.; and Ji, S. 2022. Learning Protein Representations via Complete 3D Graph Networks. *ArXiv*, abs/2207.12600.

Zemla, A. 2003. LGA: a method for finding 3D similarities in protein structures. *Nucleic acids research*, 31(13): 3370–3374.

Zhang, J.; and Zhang, Y. 2010. A novel side-chain orientation dependent potential derived from random-walk reference state for protein fold selection and structure prediction. *PloS one*, 5(10): e15386.

A Appendix

A.1 GCP Module.

As illustrated in Figure 1, GCPNET represents the features for nodes and edges within an input graph as a tuple (s, V) to distinguish scalar features (s) from vector-valued features (V). We then define $\text{GCP}_{\mathcal{F}_{ij},\lambda}(\cdot)$ to represent the GCP encoding process, where λ represents a downscaling hyperparameter (e.g., 3) and $\mathcal{F}_{ij} \in \mathbb{R}^{3\times 3}$ denotes the SO(3)-equivariant frames constructed using the **Localize** operation (i.e., the **EquiFrame** operation of (Du et al. 2022)) in Algorithm 1.

Expressing Vector Representations with V. The GCP module then expresses vector representations V as follows. The features V with representation depth r are downscaled by λ.

$$z = \{v\boldsymbol{w}_{d_z} | \boldsymbol{w}_{d_z} \in \mathbb{R}^{r\times(r/\lambda)}\} \tag{4}$$

Additionally, V is separately downscaled in preparation to be subsequently embedded as direction-sensitive edge scalar features.

$$V_s = \{v\boldsymbol{w}_{d_s} | \boldsymbol{w}_{d_s} \in \mathbb{R}^{r\times(3\times 3)}\} \tag{5}$$

Deriving Scalar Representations s'. To update scalar representations, the GCP module, in the following manner, derives two invariant sources of information from V and combines these with s:

$$q_{ij} = (V_s \cdot \mathcal{F}_{ij}) \in \mathbb{R}^9 \tag{6}$$

$$q = \begin{cases} \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} q_{ij} & \text{if } V_s \text{ represents nodes} \\ q_{ij} & \text{if } V_s \text{ represents edges} \end{cases} \tag{7}$$

$$s_{(s,q,z)} = s \cup q \cup \|z\|_2 \tag{8}$$

where $\cdot$ denotes the inner product, $\mathcal{N}(\cdot)$ represents the neighbors of a node, and $\|\cdot\|_2$ denotes the L_2 norm. Then, denote t as the representation depth of s, and let $s_{(s,q,z)} \in \mathbb{R}^{t+9+(r/\lambda)}$ with representation depth $(t + 9 + (r/\lambda))$ be projected to s' with representation depth t':

$$s_v = \{s_{(s,q,z)}\boldsymbol{w}_s + \boldsymbol{b}_s | \boldsymbol{w}_s \in \mathbb{R}^{(t+9+(r/\lambda))\times t'}\} \tag{9}$$

$$s' = \sigma_s(s_v) \tag{10}$$

Deriving Vector Representations V'. The GCP module concludes by updating vector representations as follows:

$$V_u = \{z\boldsymbol{w}_{u_z} | \boldsymbol{w}_{u_z} \in \mathbb{R}^{(r/\lambda)\times r'}\} \tag{11}$$

$$V' = \{V_u \odot \sigma_g(\sigma^+(s_v)\boldsymbol{w}_g + \boldsymbol{b}_g) | \boldsymbol{w}_g \in \mathbb{R}^{t'\times r'}\} \tag{12}$$

where $\odot$ represents element-wise multiplication and the gating function σ_g is applied row-wise to preserve SO(3) equivariance within V'.

A.2 GCPNET Layer.

The structure of a GCPNET layer follows closely after the GVP and GBP layer designs proposed in (Jing et al. 2020) and (Aykent and Xia 2022), respectively, with the following key insights and distinctions.

On Line 1 of Algorithm 1, the **Centralize** operation removes the center of mass from each node position in the input graph to ensure that such positions are subsequently 3D translation-invariant.

Thereafter, following (Du et al. 2022), the **Localize** operation on Line 2 crafts translation-invariant and SO(3)-equivariant frame encodings $\boldsymbol{\mathcal{F}}_{ij}^t = (a_{ij}^t, b_{ij}^t, c_{ij}^t)$, where $a_{ij}^t = \frac{x_i^t - x_j^t}{\|x_i^t - x_j^t\|}, b_{ij}^t = \frac{x_i^t \times x_j^t}{\|x_i^t \times x_j^t\|}$, and $c_{ij}^t = a_{ij}^t \times b_{ij}^t$, respectively. As described in more detail in Appendix A.5 and by (Du et al. 2022), these frame encodings are direction information-complete for edges, allowing networks incorporating them to effectively detect and leverage for downstream tasks the inter-atomic and force field interactions present within real-world many-body systems such as small molecules and proteins.

Line 6 represents our proposed message-passing GCP module with residual connections (**ResGCP**). Such a module is applied to iteratively update the geometric node and edge embeddings (i.e., messages) concatenated on Line 5, with residual connections between each iteration (i.e., message-passing step) and the one preceding it. Besides providing several desirable optimization-stabilizing properties (He et al. 2016), we included these residual connections as we empirically observed the tendency for standard message-passing GCP modules to produce vanishing activation sizes when applied repeatedly to update messages.

On Lines 10 and 13, we introduce a means by which to update in an SE(3)-equivariant manner the position of each node in an input 3D graph. In particular, we update node positions by residually adding learned vector-valued node features $(\chi_{v_i}^l)$ to the node positions produced by the previous GCPNET layer $(l - 1)$. As shown in Appendix A.3, such updates are initially SO(3)-equivariant, and on Line 13 we ensure these updates also become 3D translation-equivariant by adding back to each node position the input graph's original center of mass via the **Decentralize** operation. In total, this procedure produces SE(3)-equivariant updates to node positions. Additionally, for models that update node positions, we note that Line 14 updates frame encodings $\mathcal{F}_{ij}$ using the model's final predictions for node positions to provide more information-rich feature projections on Line 18 to conclude the forward pass of GCPNET.

In summary, GCPNET receives an input 3D graph $\mathcal{G}$ with node positions $\boldsymbol{x}$, scalar node and edge features, h and e, as well as vector-valued node and edge features, χ and ξ. The model is then capable of e.g., (1) predicting scalar node, edge, or graph-level properties while maintaining SE(3) invariance; (2) estimating vector-valued node, edge, or graph-level properties while ensuring SE(3) equivariance; or (3) updating node positions in an SE(3)-equivariant manner.

A.3 Proof of Proposition 1.

Proof. Suppose the vector-valued features given to the corresponding layers in GCPNET are node features χ_i and edge features ξ_{ij} that are O(3)-equivariant (i.e., 3D rotation and reflection-equivariant) by way of their construction. Additionally, suppose the scalar-valued features given to the respective layers in GCPNET are E(3)-invariant (i.e., 3D rotation, reflection, and translation-invariant) node features h_i and edge features e_{ij}. Then, following (Du et al. 2022), the **Centralize** and **Localize** operations on Lines 1-2 of Algorithm 1 will correspondingly first ensure that $\mathbf{X}^0$ is 3D translation invariant and will proceed to construct SO(3)-equivariant frames $\mathcal{F}_{ij}$. After the construction of these frames, on Line 3 all node and edge features (i.e., h_i, e_{ij}, χ_i, and ξ_{ij}) are embedded using a single **GCP** module shown.

The operations of a **GCP** module are illustrated in Figure 1, and their SO(3)-invariance for scalar features and SO(3) equivariance for vector-valued features is outlined briefly as follows. Following the proof of O(3) equivariance for the **GVP** module in (Jing et al. 2020), the proof of SO(3) equivariance within the **GCP** module is similar, with the following modifications. Within the **GCP** module, the vector-valued features (processed separately for nodes and edges) are fed not only through a bottleneck block comprised of downward and upward projection matrices $\mathbf{D_z}$ and $\mathbf{U_z}$ but are also fed into a dedicated downward projection matrix $\mathbf{D}_{\mathcal{S}}$. The output of matrix multiplication between O(3)-equivariant vector features and $\mathbf{D}_{\mathcal{S}}$ yields O(3)-equivariant vector features that are used as unique inputs for a scalarization operation. In such an operation, the dot product is taken between each O(3)-equivariant vector feature and the previously-derived SO(3)-equivariant frames $\mathcal{F}_{ij}$, yielding new SO(3)-invariant scalar features (Du et al. 2022) that are concatenated with the GCP module's remaining O(3)-invariant scalar features (i.e., L_2 norm features). Introducing SO(3)-invariant scalar information into the **GCP** module in this way breaks the 3D reflection symmetry that the previous **GVP** module enforced, giving rise to SO(3)-invariant and SO(3)-equivariant updates to scalar and vector-valued features, respectively.

In particular, the following demonstrates the invariance for our design of matrix multiplication with our **GCP** module's projection matrices (e.g., $\mathbf{D}_{\mathcal{S}}$). Suppose $\mathbf{W}_h \in \mathbb{R}^{h \times v}$, $\mathbf{V} \in \mathbb{R}^{v \times 3}$, and $\mathbf{Q} \in SO(3) \in \mathbb{R}^{3 \times 3}$. In line with (Jing et al. 2020), observe for $\mathbf{D} = (\mathbf{Q}\mathbf{V}^{\mathrm{T}}) \in \mathbb{R}^{3 \times v}$ that

$$\|\mathbf{W}_h \mathbf{D}^{\mathrm{T}}\|_2 = \|\mathbf{W}_h (\mathbf{V}^{\mathrm{T}})^{\mathrm{T}}\|_2 = \|\mathbf{W}_h \mathbf{V}\|_2.$$

Proceeding onward, Lines 4-12 of Algorithm 1 describe the operations contained within a single GCPNET layer. Via the corresponding proof in (Jing et al. 2020), by way of induction all such operations on Lines 5-9 are respectively SE(3)-invariant and SO(3)-equivariant for features $m^l_{e_{ij}}$ and $m^l_{\xi_{ij}}$. Thereby, so are features h^l_i and χ^l_i, given that the proof of equivariance for the equivariant **LayerNorm** and **Dropout** operations employed within the **GCP** module has previously been concretized by (Jing et al. 2020). Lines 10-12 conclude the operations of a GCPNET layer by potentially updating the 3D positions of each node i in the input 3D graph. To do so, GCPNET residually updates current node positions using SO(3)-equivariant vector-valued features $\chi^l_{v_i}$.

Lastly, Lines 13-15 add back the original position of the centroid of the input 3D graph to each new node position in the graph, ultimately imbuing position updates within $\mathbf{X}^l$ with SE(3) equivariance. Line 18 then concludes by performing a final SO(3)-invariant and SO(3)-equivariant projection for scalar and vector-valued features, respectively. Therefore, GCPNETS are SE(3)-invariant for scalar feature updates, SE(3)-equivariant for vector-valued node position and feature updates, and, as a consequence, satisfy the constraint proposed in Eq. 1.

□

A.4 Proof of Proposition 2.

Proof. The proof of SE(3) invariance for scalar node and edge features, h_i and e_{ij}, follows as a corollary of Appendix A.3. Therefore, GCPNETS are SE(3)-invariant concerning their predicted scalar node and edge features and, as a consequence, are geometrically complete according to the constraint in Eq. 2.

□

A.5 Proof of Proposition 3.

Proof. Suppose that GCPNET designates its local geometric representation for layer t to be $\boldsymbol{\mathcal{F}}^t_{ij} = (a^t_{ij}, b^t_{ij}, c^t_{ij})$, where $a^t_{ij} = \frac{x^t_i - x^t_j}{\|x^t_i - x^t_j\|}$, $b^t_{ij} = \frac{x^t_i \times x^t_j}{\|x^t_i \times x^t_j\|}$, and $c^t_{ij} = a^t_{ij} \times b^t_{ij}$, respectively. In (Du et al. 2022), this SO(3)-equivariant formulation of $\boldsymbol{\mathcal{F}}^t_{ij}$ is proven to be a local orthonormal basis at the tangent space of x^t_i and is thereby geometrically complete, allowing no loss of geometric information as shown in Appendix A.5 of (Du et al. 2022). Therefore, GCPNETS are geometrically complete.

□

A.6 Implementation Details.

Featurization. For the LBA and PSR tasks, in each 3D input graph, we include as a scalar node feature an atom's type using a 9-dimensional one-hot encoding vector for each

Table 4: Summary of GCPNET's node and edge features for 3D input graphs derived for the LBA and PSR tasks. Here, N and E denote the number of nodes and edges in $\mathcal{G}$, respectively.

	Feature	Type	Shape
Node Features (h)	One-hot encoding of atom type	Categorical (Scalar)	$N \times 9$
Node Features (χ)	Directional encoding of orientation	Numeric (Vector)	$N \times 2$
Edge Features (e)	Radial basis distance embedding	Numeric (Scalar)	$E \times 16$
Edge Features (ξ)	Pairwise atom position displacement	Numeric (Vector)	$E \times 1$
Total	Node features		$N \times 11$
	Edge features		$E \times 17$

Table 5: Summary of GCPNET's node and edge features for 3D input graphs derived for the NMS task.

	Feature	Type	Shape
Node Features (h)	Invariant velocity encoding	Numeric (Scalar)	$N \times 1$
Node Features (χ)	Velocity and orientation encoding	Numeric (Vector)	$N \times 3$
Edge Features (e)	Edge and distance embedding	Numeric (Scalar)	$E \times 17$
Edge Features (ξ)	Pairwise atom position displacement	Numeric (Vector)	$E \times 1$
Total	Node features		$N \times 4$
	Edge features		$E \times 18$

atom. As vector-valued node features, we include *forward* and *reverse* unit vectors in the direction of $x_{i+1} - x_i$ and $x_{i-1} - x_i$, respectively (i.e., the node's 3D orientation). For the input 3D graphs' scalar edge features, we encode the distance $\|x_i - x_j\|_2$ using Gaussian radial basis functions, where we use 16 radial basis functions with centers evenly distributed between 0 and 20 units (e.g., Angstrom). For the graphs' vector-valued edge features, we encode the unit vector in the direction of $x_i - x_j$ (i.e., pairwise atom position displacements).

For the NMS task, in each 3D input graph, we include as a scalar node feature an invariant encoding of each node's velocity vector, namely $\sqrt{v_i^2}$. Each node's velocity and orientation are encoded as vector-valued node features. Scalar edge features are represented as Gaussian radial basis distance encodings as well as the product of the charges in each node pair (i.e., $c_i c_j$). Lastly, vector-valued edge features are represented as pairwise atom position displacements.

Hardware Used. The Oak Ridge Leadership Facility (OLCF) at the Oak Ridge National Laboratory (ORNL) is an open science computing facility that supports HPC research. The OLCF houses the Summit compute cluster. Summit, launched in 2018, delivers 8 times the computational performance of Titan's 18,688 nodes, using only 4,608 nodes. Like Titan, Summit has a hybrid architecture, and each node contains multiple IBM POWER9 CPUs and NVIDIA Volta GPUs all connected with NVIDIA's high-speed NVLink. Each node has over half a terabyte of coherent memory (high bandwidth memory + DDR4) addressable by all CPUs and GPUs plus 800GB of non-volatile RAM that can be used as a burst buffer or as extended memory. To provide a high rate of I/O throughput, the nodes are connected in a non-blocking fat-tree using a dual-rail Mellanox EDR InfiniBand interconnect. We used the Summit compute cluster to train all our models. For the LBA and NMS tasks, we used 16GB NVIDIA Tesla V100 GPUs for model training, whereas for the memory-intensive PSR task, we used 32GB V100 GPUs instead.

Software Used. We used Python 3.8.12 (Van Rossum and Drake 2009), PyTorch 1.10.2 (Paszke et al. 2019), PyTorch Lightning 1.7.7 (Falcon 2019), and PyTorch Geometric 2.1.0post0 (Fey and Lenssen 2019) to run our deep learning experiments. For each model trained, PyTorch Lightning was used to facilitate model checkpointing, metrics reporting, and distributed data parallelism across 6 V100 GPUs. A more in-depth description of the software environment used to train and run inference with our models can be found at https://github.com/BioinfoMachineLearning/GCPNet.

Hyperparameters. We use a learning rate of 10^{-4} for all GCPNET models. The learning rate is kept constant throughout each model's training. Each model is trained for a minimum of 100 epochs and for a maximum of 12,000 epochs for the NMS task and for a maximum of 1,000 epochs for all other tasks, respectively. For a given task, models with the best loss on the corresponding validation data split are then tested on the test split for the respective task.

Table 6: Hyperparameter search space for all GCPNET models through which we searched to obtain strong performance on the LBA task's validation split. The final parameters for the standard GCPNET model for the LBA task are in **bold**.

Hyperparameter	Search Space
Number of GCPNET Layers	7, **8**
Number of GCP Message-Passing Layers	**8**
χ Hidden Dimensionality	**16**, 32
Learning Rate	**0.0001**, 0.0003
Weight Decay Rate	**0**
GCP Dropout Rate	**0.1**, 0.25
Dense Layer Dropout Rate	**0.1**, 0.25

Table 7: Hyperparameter search space for all GCPNET models through which we searched to obtain strong performance on the PSR task's validation split. The final parameters for the standard GCPNET model for the PSR task are in **bold**.

Hyperparameter	Search Space
Number of GCPNET Layers	**5**
Number of GCP Message-Passing Layers	**8**
χ Hidden Dimensionality	**16**, 32
Learning Rate	**0.0001**, 0.0003
Weight Decay Rate	**0**, 0.0001
GCP Dropout Rate	**0.1**, 0.25
Dense Layer Dropout Rate	**0.1**, 0.25

Table 8: Hyperparameter search space for all GCPNet models through which we searched to obtain strong performance on the NMS task's validation split. The final parameters for the standard GCPNet model for the NMS task are in **bold**.

Hyperparameter	Search Space
Number of GCPNet Layers	**4**, 7
Number of GCP Message-Passing Layers	**8**
χ Hidden Dimensionality	**16**
Learning Rate	**0.0001**, 0.0003
Weight Decay Rate	**0**
GCP Dropout Rate	0.0, **0.1**