Equivariant graph neural networks as surrogate for computational fluid dynamics in 3D artery models

Julian Suk
Department of Applied Mathematics
University of Twente
Enschede, The Netherlands
j.m.suk@utwente.nl

Pim de Haan
QUVA Lab, University of Amsterdam
Qualcomm AI Research
Qualcomm Technologies Netherlands B.V.

Phillip Lippe
QUVA Lab
University of Amsterdam
Amsterdam, The Netherlands

Christoph Brune
Department of Applied Mathematics
University of Twente
Enschede, The Netherlands

Jelmer M. Wolterink
Department of Applied Mathematics
& Technical Medical Center
University of Twente
Enschede, The Netherlands

Abstract

Computational fluid dynamics (CFD) is an invaluable tool in modern physics but the time-intensity and computational complexity limit its applicability to practical problems, e.g. in medicine. Surrogate methods could speed up inference and allow for use in such time-critical applications. We consider the problem of estimating hemodynamic quantities (i.e. related to blood flow) on the surface of 3D artery geometries and employ anisotropic graph convolution in an end-to-end $\mathrm{SO}(3)$-equivariant neural network operating directly on the polygonal surface mesh. We show that our network can accurately predict hemodynamic vectors for each vertex on the surface mesh with normalised mean absolute error of 0.6 [%] and approximation accuracy of 90.5 [%], demonstrating its feasibility as surrogate method for CFD.

1 Introduction

Computational fluid dynamics (CFD) has been successfully used in many fields of physical science. Advances have been made to leverage the powerful tool to gain insight into blood flow in human arteries and provide doctors with hemodynamic biomarkers to aid them in patient treatment. For instance, arterial wall shear stress (WSS) has been found to correlate with plaque development and arterial remodelling [13, 7] which can lead to atherosclerosis and ultimately, death, but can in some cases be prevented by e.g. stent placement. To inform appropriate treatment, doctors are however rarely interested in overly precise hemodynamic quantities but usually seek qualitative evaluation. Slightly lower accuracy might be acceptable at greatly increased efficiency for many practical purposes. Thus, clinical practice could benefit greatly from surrogate methods for the time-consuming CFD.

Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021).

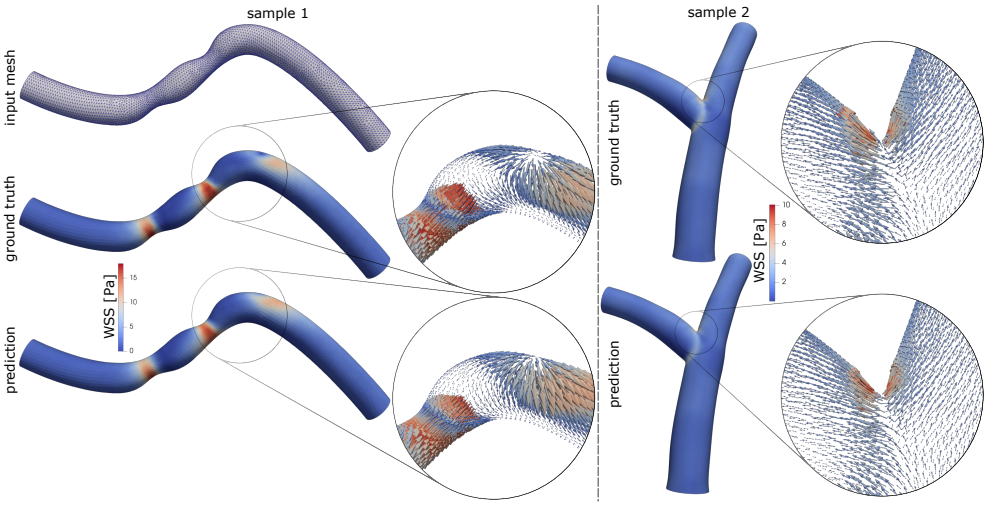

Figure 1: **Sample predictions** of a gauge-equivariant mesh graph convolutional network (GEM-GCN) on unseen geometries from the single artery and bifurcation datasets. Ground truth obtained by CFD. Given an input surface mesh, the GCN predicts vector-valued WSS for each mesh vertex.

CFD in arterial geometries require the extraction of polygon meshes from medical images [9, 10]. Our key observation is that every mesh consists of vertices $\mathcal{V}$ and faces $\mathcal{F}$ which induce a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and that this graph is naturally suited for use with graph neural networks. However, previous work using deep learning to predict hemodynamic quantities has employed hand-crafted parametrisation of the artery surface [8, 14] to match conventional convolutional neural networks. Supporting our intuition, Morales et al. [4] recently showed that GCNs outperfom previous approaches for the prediction of scalar hemodynamic potentials in the left atrial appendage.

While widely used message passing graph convolutional networks (GCN) like GraphSAGE [6] are generally isotropic, recent works propose more expressive anisotropic convolution filters [2, 3]. Furthermore, in geometric deep learning [1] symmetry considerations have been used to boost the performance of deep neural networks. These allow the models to make **equivariant** predictions that adapt to spatial orientation of the input, removing the need for shape registration in arterial geometries and boosting data efficiency. In medicine, patient data is notoriously hard to come by and efficiently making use of it is crucial. Leveraging these recent advances, we design an anisotropic GCN to be equivariant under $\mathrm{SO}(3)$ transformation, i.e. rotation in 3D Euclidean space. We apply our model to two datasets of synthetic coronary arteries and train it to predict vector-valued hemodynamic quantities by regression on the ground truth obtained from CFD.

2 Method

We cascade graph convolution and pooling layers in a three-scale U-Net [12] residual architecture which is depicted in Figure 2. These models are trained using regression on ground truth labels obtained by CFD to predict vector-valued quantities on the vertices of an input mesh.

2.1 Anisotropic message passing

In order to realise graph convolution via message passing with **anisotropic** convolution filters, we employ gauge-equivariant mesh convolution layers [3] on the mesh graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$:

$$((K^1, K^2(\cdot,\cdot)) * f)_p := f_p \cdot K^1 + \sum_{q \in N(p)} \rho(p,q) f_q \cdot K^2(p,q), \qquad p \in \mathcal{V} \tag{1}$$

where $K^1, K^2(p,q) \in \mathbb{R}^{c_i \times c_{i+1}}$, $f : \mathcal{V} \to \mathbb{R}^{c_i}$, $f_p = f(p)$, $p \in \mathcal{V}$, $\rho(p,q) \in \mathbb{R}^{c_i \times c_i}$, and $N(p)$ is a neighbourhood around p that determines the filter support. The kernels K^1, K^2 carry trainable weights, f_p is the feature vector associated with vertex p and $\rho(p,q)$ parallel-transports feature vectors across the mesh surface for geometrically valid linear combination.

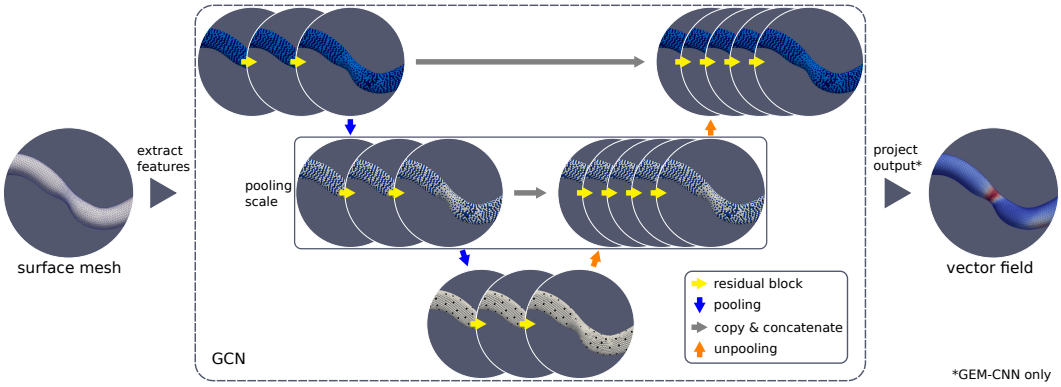

Figure 2: **Network architecture.** Non-gray vertices are used for message passing on each pooling scale. Residual blocks consist of two convolution layers and skip connection.

Definition 1 (Anisotropy)**.** We call a function $F(p, q)$, $q \in N(p)$ anisotropic if it is not constant in q. Consequently, we call a layer anisotropic, if it contains any anisotropic function.

Gauge-equivariant mesh convolution requires expressing $f \in \mathcal{X}(\mathcal{V}, \mathbb{R}^{c_i})$ as a collection of tangential signals on the mesh surface and restricting the kernel $K^2(p, q)$ so the operation is equivariant to the tangential gauges.

In the context of graph convolution on meshes we can recover two other popular forms of message passing from equation (1) which are also implemented in PyTorch Geometric [5] ("PyG"). Specifically, picking an **isotropic** kernel $K^2(p, q) = \frac{1}{|N(p)|}\bar{K}^2$ and isotropic $\rho(p, q) = \mathrm{I} \in \{0, 1\}^{c_i \times c_i}$ leads to GraphSAGE convolution [6] while using an anisotropic, learned neighbourhood attention mechanism $\rho(p, q) = \sigma(w \cdot (f_q - f_p))\mathrm{I}$ results in feature-steered (FeaSt) convolution [15]. We compare performance of the three message passing algorithms in our experiments.

2.2 $\mathrm{SO}(3)$ equivariance

Gauge-equivariant mesh convolution layers preserve $\mathrm{SO}(3)$ equivariance, i.e. any rotation of an equivariant input vector field in 3D will propagate to a corresponding rotation at the output. This is because they define message passing and feature vectors in tangential planes to the object surface, which rotate with the input mesh by construction. When cascading layers into a network architecture, end-to-end $\mathrm{SO}(3)$ equivariance can be achieved by making sure the network's input features are by construction $\mathrm{SO}(3)$-equivariant. We use linear combinations of relative vertex position and normal differences as input features. Additionally, we supply the network with a sense of flow direction by vertex-wise geodesic distances to the artery inlet.

2.3 Pooling

Graph pooling consists of graph clustering followed by feature-vector collapsing. We create a hierarchy of $n + 1$ vertex subsets $\mathcal{V}^0 \supset \mathcal{V}^1 \supset \cdots \supset \mathcal{V}^n$ via farthest point sampling and determine clusters by nearest geodesic neighbours. Formally, the pooling operator ψ_i to level i with clusters $C(p)$ for $p \in \mathcal{V}^i$ is

$$\psi_i : \begin{cases} \mathcal{X}(\mathcal{V}^{i-1}, \mathbb{R}^c) \to \mathcal{X}(\mathcal{V}^i, \mathbb{R}^c) \\ f_p \mapsto \frac{1}{|C(p)|} \sum\limits_{q \in C(p)} \rho(p, q) f_q, \qquad p \in \mathcal{V}^i \end{cases} \tag{2}$$

Unpooling $\hat{\psi}$ simply copies feature vectors f_p back to each cluster element $q \in C(p)$.

$$\hat{\psi}_i : \begin{cases} \mathcal{X}(\mathcal{V}^i, \mathbb{R}^c) \to \mathcal{X}(\mathcal{V}^{i-1}, \mathbb{R}^c) \\ f_q \mapsto \rho(p, q)^{-1} f_p, \qquad q \in C(p), \qquad p \in \mathcal{V}^{i-1} \end{cases} \tag{3}$$

		NMAE [%]			ε [%]			$\triangle_{\text{max}}$ [Pa]			$\triangle_{\text{mean}}$ [Pa]		
		mean	median	75th	mean	median	75th	mean	median	75th	mean	median	75th
Single Arteries $L_{\text{max}} = 21.86$ [Pa] $L_{\text{median}} = 2.03$ [Pa]	SAGE-GCN	2.2	2.0	2.6	32.4	30.0	37.0	10.41	7.80	14.65	1.11	1.01	1.32
	FeaSt-GCN	1.2	1.1	1.5	19.0	18.6	22.4	5.83	5.13	8.17	0.60	0.57	0.77
	GEM-GCN	**0.6**	**0.6**	**0.8**	**9.9**	**9.5**	**11.6**	**3.94**	**3.68**	**5.46**	**0.32**	**0.31**	**0.41**
	SAGE-GCN†	10.5	9.6	12.8	149.2	128.1	181.2	26.73	23.96	36.17	5.31	4.84	6.50
	FeaSt-GCN†	8.3	7.5	10.1	123.7	111.1	152.9	25.63	22.93	34.52	4.22	3.82	5.13
	GEM-GCN†	**0.6**	**0.6**	**0.8**	**9.8**	**9.4**	**11.4**	**3.80**	**3.39**	**5.53**	**0.32**	**0.31**	**0.42**
Bifurcating Arteries $L_{\text{max}} = 7.16$ [Pa] $L_{\text{median}} = 1.27$ [Pa]	SAGE-GCN	1.3	1.1	1.5	24.4	21.1	27.3	4.38	4.14	5.50	0.27	0.22	0.29
	FeaSt-GCN	**1.2**	**0.9**	**1.3**	**20.7**	**18.1**	**22.3**	4.10	3.72	4.77	**0.23**	**0.19**	**0.25**
	GEM-GCN	1.3	1.1	1.6	23.3	20.4	28.6	**3.99**	**3.71**	**4.64**	0.27	0.22	0.32
	SAGE-GCN†	7.3	6.9	9.6	117.4	115.2	151.3	8.60	8.29	10.10	1.45	1.37	1.91
	FeaSt-GCN†	7.4	7.4	10.1	119.6	117.1	161.1	8.39	8.33	9.78	1.48	1.47	2.01
	GEM-GCN†	**1.3**	**1.1**	**1.6**	**23.3**	**20.9**	**28.5**	**4.00**	**3.67**	**4.65**	**0.26**	**0.22**	**0.32**

†evaluated on randomly rotated test samples

Table 1: **Prediction accuracy** on held-out test splits for two datasets of synthetic coronary arteries with single output and bifurcation. Ground truth labels are obtained via CFD.

On each pooling scale $p \in \mathcal{V}^i$ we define neighbourhoods $N(p)$ based on radius r in Euclidean space so that

$$N(p) = \{q \in \mathcal{V}^i | \|p - q\|_2 < r\}, \qquad p \in \mathcal{V}^i \tag{4}$$

Subsequently we connect p to all $q \in N(p)$ which results in a hierarchy of graphs $\mathcal{G}^i = (\mathcal{V}^i, \mathcal{E}^i)$. The constructed neighbourhoods $N(p)$ define convolution filters with fixed spatial support. Because of this, we are able to process input meshes with heterogeneous mesh size. For pooling and unpooling to be applicable for FeaSt convolution architectures we choose $\rho(p, q) = \rho(p, q)^{-1} = \mathrm{I}$ in equations (2) & (3). Our implementation is available online. [1]

3 Numerical experiments

We train GCNs consisting of the three convolution types by mean-squared-error regression using Adam optimiser on two datasets of 2×10^3 synthetic coronary artery geometries each with ground truth labels obtained via CFD. For further reference we call these SAGE-GCN, FeaSt-GCN and GEM-GCN. The datasets consist of arteries with single inlet and outlet (ca. 8×10^3 vertices & 17×10^3 faces) or bifurcating arteries splitting in two (ca. 17×10^3 vertices & 32×10^3 faces) while our models can be applied to either. The bifurcation dataset is randomly generated from empirical shape distributions [11]. All artery geometries are used to generate a domain for the solution of incompressible Navier-Stokes equations which are solved by CFD. Blood flow boundary conditions are fixed across either dataset. With blood velocity u, dynamic viscosity μ, and surface normal $\vec{n}$, WSS is defined $\tau = \mu(\nabla u \cdot \vec{n})$ on the artery surface.

We argue that WSS depends in good approximation predominantly on the lumen wall topology and flow boundary conditions latter of which are implicitly encoded in our datasets. This is especially valid for arteries where the vessel wall is mainly responsible for guiding the blood flow.

SAGE-GCN and FeaSt-GCN are trained on an NVIDIA GeForce RTX 3080 10 GB and GEM-GCN on two NVIDIA Quadro RTX 6000 24 GB. Training times range from 3:00 [h] (SAGE-GCN & single arteries) to 55:00 [h] (GEM-GCN & bifurcations) depending on the complexity of model and dataset. After training, inference takes less than 5 [s] per shape including pre-processing. Sample predictions are shown in Figure 1.

As error metrics, we report normalised mean absolute error (NMAE) and define an approximation error $\varepsilon := \|\triangle\|_2 / \|L\|_2$. Elements of $\triangle$ are vertex-wise 2-normed differences between the network output $f^{\text{out}} \in \mathcal{X}(\mathcal{V}, \mathbb{R}^3)$ and ground truth label $l \in \mathcal{X}(\mathcal{V}, \mathbb{R}^3)$ so that $\triangle_p = \|f^{\text{out}}(p) - l(p)\|_2$ for $p \in \mathcal{V}$ and $L_p = \|l_p\|_2$. Additionally, we report the maximum and mean vertex-wise normed difference, i.e. $\triangle_{\text{max}} = \max_p\{\triangle_p\}$ and $\triangle_{\text{mean}} = (\sum_p \triangle_p)/|\mathcal{V}|$. For scale we provide the median of the label statistics $L_{\text{max}} = \max_p \|L_p\|$ and $L_{\text{median}} = \text{median}_p \|L_p\|$ over the test sets.

[1] `https://github.com/sukjulian/coronary-mesh-convolution`

Results are shown in Table 1. The presented error metrics suggest strictly better performance on the single artery dataset when using anisotropic convolution and pooling. On the bifurcating artery dataset the difference in performance is less pronounced, suggesting that the choice of input feature descriptor lacks expressiveness for the task at hand. We argue that being able to work with direction should always benefit a GCN, unless the supplied input feature, transformed into the network's internal feature representation hinders inference. In this case, a more efficient proxy for the task might be found by a less restricted GCN. We check all networks for $\mathrm{SO}(3)$ equivariance by randomly rotating input samples at inference time. Rotation has a dramatic effect on the predicted WSS vectors when SAGE-GCN or FeaSt-GCN are used, but not when GEM-GCN is used, quantitatively confirming its $\mathrm{SO}(3)$ equivariance.

4 Discussion

We demonstrate how graph convolutional neural networks operating on meshes can be used as surrogate method for computational fluid dynamics in time-critical applications. We employ novel anisotropic graph convolution and our networks are equivariant under $\mathrm{SO}(3)$ transformation, leading to improved performance and data efficiency.

The biggest limitation of our current method is the independence of boundary conditions. Boundary conditions are a crucial element of CFD and we will investigate how we can condition GCNs in this respect. Furthermore, quantifying prediction uncertainty seems like a logical next step. It remains to be examined to what extent graph convolution on meshes via message passing generalises to different discretisations of the surface mesh.

Societal impact

In general, the performance of machine learning algorithms strongly depends on their training data. Thus is it imperative to guarantee a balanced dataset in case the algorithm will be applied in a way that affects humans. Furthermore, surrogate methods introduce an error and if clinical decisions should be based on them, they must provide a confidence score and quantify uncertainty.

Acknowledgments

This work is funded in part by the 4TU Precision Medicine programme supported by High Tech for a Sustainable Future, a framework commissioned by the four Universities of Technology of the Netherlands. Jelmer M. Wolterink was supported by the NWO domain Applied and Engineering Sciences VENI grant (18192).

References

[1] Michael Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: grids, groups, graphs, geodesics, and gauges. *ArXiv*, 2021.

[2] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral CNN. In *ICML*, volume 97, 2019.

[3] Pim de Haan, Maurice Weiler, Taco Cohen, and Max Welling. Gauge equivariant mesh CNNs: anisotropic convolutions on geometric graphs. In *ICLR*, 2021.

[4] Xabier Morales Ferez, Jordi Mill, Kristine A. Juhl, Cesar Acebes, Xavier Iriart, Benoit Legghe, Hubert Cochet, Ole de Backer, Rasmus R. Paulsen, and Oscar Camara. Deep learning framework for real-time estimation of in-silico thrombotic risk indices in the left atrial appendage. *Front. Physiol.*, 12, 2021.

[5] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[6] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[7] Ayla Hoogendoorn, Annette Kok, Eline M. J. Hartman, Giuseppe de Nisco, Lorena Casadonte, Claudio Chiastra, Adriaan Coenen, Suze-Anne Korteland, Kim van der Heiden, Frank Gijsen, Dirk Duncker, Antonius V. D. van der Steen, and Jolanda Wentzel. Multidirectional wall shear stress promotes advanced coronary plaque development: comparing five shear stress metrics. *Cardiovasc. Res.*, 116, 2019.

[8] Lucian M. Itu, Sai Rapaka, Tiziano Passerini, Bogdan Georgescu, Chris Schwemmer, Max Schoebinger, Thomas Flohr, Puneet Sharma, and Dorin Comaniciu. A machine learning approach for computation of fractional flow reserve from coronary computed tomography. *J. Appl. Physiol.*, 121, 2016.

[9] David Lesage, Elsa D. Angelini, Isabelle Bloch, and Gareth Funka-Lea. A review of 3D vessel lumen segmentation techniques: Models, features and extraction schemes. *Med. Image Anal.*, 13(6), 2009.

[10] Geert Litjens, Francesco Ciompi, Jelmer M. Wolterink, Bob D. de Vos, Tim Leiner, Jonas Teuwen, and Ivana Išgum. State-of-the-art deep learning in cardiovascular image analysis. *JACC: Cardiovasc. Imaging*, 12(8, Part 1), 2019.

[11] Pau Medrano-Gracia, John Ormiston, Mark Webster, Susann Beier, Alistair Young, Chris Ellis, Chunliang Wang, Örjan Smedby, and Brett Cowan. A computational atlas of normal coronary artery anatomy. *EuroIntervention : journal of EuroPCR in collaboration with the Working Group on Interventional Cardiology of the European Society of Cardiology*, 12(7), 2016.

[12] Olaf Ronneberger, Philipp Fischer, Thomas Brox, Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. U-Net: convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[13] Habib Samady, Parham Eshtehardi, Michael Mcdaniel, Jin Suo, Saurabh Dhawan, Charles Maynard, Lucas Timmins, Arshed Quyyumi, and Don Giddens. Coronary artery wall shear stress is associated with progression and transformation of atherosclerotic plaque and arterial remodeling in patients with coronary artery disease. *Circulation*, 124, 2011.

[14] Boyang Su, Jun-Mei Zhang, Hua Zou, Dhanjoo Ghista, Thu Thao Le, and Calvin Chin. Generating wall shear stress for coronary artery in real-time using neural networks: feasibility and initial results based on idealized models. *Comput. Biol. Med.*, 126, 2020.

[15] Nitika Verma, Edmond Boyer, and Jakob Verbeek. FeaStNet: feature-steered graph convolutions for 3D shape analysis. In *CVPR*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [No] **but all the details are in the code on GitHub**
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] **Table 1**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] **PyTorch Geometric**
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] **code on GitHub**
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] **only synthetic data used**
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]