Any-Subgroup Equivariant Networks via Symmetry Breaking

Abhinav Goel AMGOEL@MIT.EDU
Derek Lim DEREKLIM@MIT.EDU
Hannah Lawrence HANLAW@MIT.EDU
MIT

Stefanie Jegelka STEFJE@MIT.EDU
TUM and MIT

Ningyuan Huang THUANG@FLATIRONINSTITUTE.ORG
Flatiron Institute

Editors: List of editors' names

Abstract

The inclusion of symmetries as an inductive bias, known as "equivariance", often improves generalization on geometric data (e.g. grids, sets, and graphs). However, equivariant architectures are usually highly constrained, designed for symmetries chosen *a priori*, and not applicable to datasets with other symmetries. This precludes the development of flexible, multi-modal foundation models capable of processing diverse data equivariantly. In this work, we build a single model — the Any-Subgroup Equivariant Network (ASEN) — that can be simultaneously equivariant to several groups, simply by modulating a certain auxiliary input feature. In particular, we start with a fully permutation-equivariant base model, and then obtain subgroup equivariance by using a symmetry-breaking input whose automorphism group is that subgroup. However, finding an input with the desired automorphism group is computationally hard. We overcome this by relaxing from exact to approximate symmetry breaking, leveraging the notion of 2-closure to derive fast algorithms. Theoretically, we show that our subgroup-equivariant networks can simulate equivariant MLPs, and their universality can be guaranteed if the base model is universal. Empirically, we validate our method on symmetry selection, multitask, and transfer learning settings, demonstrating that a single network equivariant to multiple permutation subgroups outperforms both separate equivariant models and a single non-equivariant model.

Keywords: Equivariance, symmetry breaking, graph neural networks, graph automorphism

1. Introduction

Equivariant machine learning exploits symmetries in data to constrain the model with known priors, often leading to improved generalization [10; 22], interpretability [8], and efficiency [6]. Most existing equivariant models are tailored to specific symmetry groups chosen *a priori*, such as graph neural networks (GNNs) and DeepSets. While these equivariant models have demonstrated promising performance in datasets satisfying the prescribed symmetries, they are *inflexible* in several important ways: (I) equivariant architectures typically require deriving and implementing group-specific equivariant layers, so substantial research and

engineering must be done for architectural design whenever a new type of symmetry arises, and (II) equivariant architectures are typically only equivariant to one symmetry group, and significantly differ between domains. Thus, an equivariant model cannot easily learn from or make predictions across domains with distinct symmetries, so equivariant models cannot benefit from the empirical successes of the foundation model paradigm [9].

In this work, we introduce a framework for building *flexible* equivariant networks, named *Any-Subgroup Equivariant Networks* (ASEN). Given a subgroup $G \leq \mathbf{G}$, ASEN parameterizes G-equivariant functions via a base $\mathbf{G}$-equivariant model, and a symmetry breaking object $\mathbf{v}$ whose autormophism group is G. A trivial, but widespread, example of this technique is the use of positional encodings [27] to fully break the permutational symmetry of transformers (since every entry of the positional encoding vector is unique, $G = \mathrm{Aut}(\mathbf{v})$ is the trivial group). When $\mathbf{v}$ has non-trivial automorphism, *some* equivariance is retained.

ASEN overcomes inflexibility (I), since it only requires providing a single new input $\mathbf{v}$ (with correct automorphism group) to a base network h_θ. Also, ASEN overcomes inflexibility (II), since a single instance of our model can process data from varying domains with different symmetry groups.

While our framework works for any $\mathbf{G}$, we focus on the particular case when $\mathbf{G} = S_n$ is the symmetry group acting as permutation matrices, so that our networks are equivariant to permutation subgroups. This covers the symmetries of many common domains, and allows us to leverage existing permutation equivariant models as the base model h_θ. In particular, we may leverage existing set networks (with inputs in $\mathbb{R}^n$), graph neural networks (inputs in $\mathbb{R}^{n^2}$) and hypergraph neural networks (inputs in $\mathbb{R}^{n^K}$) for the base model. While large K may be required for complex groups, to balance efficiency and expressivity, we focus on the $K = 2$ case of graph neural networks, and develop a practical algorithm for computing edge features $\mathbf{v} \in \mathbb{R}^{n^2}$ with (nearly) the desired self-symmetry, $\mathrm{Aut}(\mathbf{v}) \approx G$. To do this, we use the notion of the 2-closure $G^{(2)}$ of a group G [23], which provides a formal notion of a group that is close to the target group ($G^{(2)} \approx G$), and we compute $\mathbf{v}$ with $\mathrm{Aut}(\mathbf{v}) = G^{(2)}$.

Theoretically, we show that ASEN exactly enforces the desired symmetries via proper choices of the symmetry-breaking input and architecture (Proposition 1, Lemma 2). We also prove that ASEN is as expressive as certain equivariant MLPs (Theorem 3), with universality guarantees given a sufficiently expressive base model (Theorem 4). Empirically, we demonstrate that ASEN with a single architecture can match performance using separate equivariant models in graph learning (Appendix D.1) or non-equivariant baselines for sequence tasks (Section 3), highlighting its ability to transfer structural knowledge across diverse tasks.

2. Method

General method (ASEN) Here, we describe our general framework for ASEN. Let $\mathbf{G}$ be a matrix group, and let $G \leq \mathbf{G}$ be a subgroup. Both groups act on the sets $\mathcal{X}$ and $\mathcal{Y}$. We desire our model to parameterize G-equivariant functions, that is, functions $f : \mathcal{X} \to \mathcal{Y}$ such that $f(gx) = gf(y)$ for $g \in G$. To this end, we consider a "lift" of f: a function $h_\theta : \mathcal{X} \times \mathcal{V} \to \mathcal{Y}$ on an expanded space, where we introduce an additional space $\mathcal{V}$ on which

G and $\mathbf{G}$ act. The function h_θ is $\mathbf{G}$-equivariant (i.e. equivariant to the larger group):

$$h_\theta(gx, gv) = gh_\theta(x, v), \quad g \in \mathbf{G}. \tag{1}$$

To obtain f_θ from h_θ, we find a symmetry-breaking input $\mathbf{v} \in \mathcal{V}$ that is exactly self-symmetric to the subgroup G, i.e. its automorphism group is G,

$$\mathrm{Aut}(\mathbf{v}) = \{g \in \mathbf{G} : g\mathbf{v} = \mathbf{v}\} = G. \tag{2}$$

Finally, we define our G-equivariant model $f_\theta : \mathcal{X} \to \mathcal{Y}$ as

$$f_\theta(x) = h_\theta(x, \mathbf{v}). \tag{3}$$

The model f_θ is G-equivariant because for any $g \in G$,

$$f_\theta(gx) = h_\theta(gx, \mathbf{v}) = h_\theta(gx, g\mathbf{v}) = gh_\theta(x, \mathbf{v}) = gf_\theta(x), \tag{4}$$

where the second equality follows from $g \in \mathrm{Aut}(\mathbf{v})$, and the third equality is due to $\mathbf{G}$-equivariance of h_θ. In fact, if h_θ satisfies an injectivity condition, then f_θ is *only* equivariant to G, and not to any other elements in the larger group $\mathbf{G}$, as shown in Proposition 1 below.

Proposition 1 *Let $h_\theta : \mathcal{X} \times \mathcal{V} \to \mathcal{Y}$ be $\mathbf{G}$-equivariant, and let $\mathrm{Aut}(\mathbf{v}) = G$. Then $f_\theta(x) := h_\theta(x, \mathbf{v})$ is equivariant to G. If additionally h_θ is injective in the input $\mathbf{v}$, then f_θ is not equivariant to any transformation in $\mathbf{G} \setminus G$.*

Permutation Subgroup Equivariance via Hypergraph Symmetry Breaking To use our ASEN framework for parameterizing a G-equivariant function, we need two main components: (i) a method of parameterizing the base model h_θ that is equivariant to the larger group $\mathbf{G}$, and (ii) a way to construct or compute a symmetry breaking object $\mathbf{v}$ with automorphism group $\mathrm{Aut}(\mathbf{v}) = G$. In this subsection, we show that when $\mathbf{G} = S_n$ is the symmetric group acting as permutation matrices on n objects, we can leverage existing equivariant architectures for (i); and we can develop a practical algorithm for (ii). For the rest of this paper, we primarily focus on this setting.

To construct efficient and expressive symmetry breaking objects, we turn to *hypergraphs*. Concretely, for a (matrix) group G acting on $\mathbb{R}^n$, a hypergraph on n nodes is defined as $\mathcal{H} = (A^{(1)}, A^{(2)}, \ldots, A^{(K)})$, where $A^{(k)} \in \mathbb{R}^{n^k}$ is an order k-tensor, and K is the max tensor order. We can interpret $A^{(1)}$ as a (node) positional encoding, and $A^{(k)}$ as (hyper-)edge features for $k \geq 2$. The *automorphism group* of the hypergraph is defined as

$$\mathrm{Aut}(\mathcal{H}) = \{P \in S_n : P^{\otimes k} A^{(k)} = A^{(k)}, k = 1, \ldots, K\}. \tag{5}$$

For instance, if $K = 2$, then this is the standard graph automorphism group

$$\mathrm{Aut}(\mathcal{H}) = \{P \in S_n : PA^{(1)} = A^{(1)}, PA^{(2)}P^\top = A^{(2)}\}. \tag{6}$$

For K large enough, we can construct a hypergraph $\mathcal{H}$ such that $\mathrm{Aut}(\mathcal{H})$ uniquely determines G [30]. Then, we consider any existing permutation-equivariant hypergraph neural network $h_\theta : \mathbb{R}^n \times \prod_{k=1}^{K} \mathbb{R}^{n^k} \to \mathbb{R}^n$ such that any $P \in S_n$,

$$h_\theta(PX, PA^{(1)}, \ldots, P^{\otimes^K} A^{(K)}) = P h_\theta(X, A^{(1)}, \ldots, A^{(K)}). \tag{7}$$

We abbreviate (7) by $h_\theta(P(X, \mathcal{H})) = P h_\theta(X, \mathcal{H})$. Our G-equivariant model f then takes the form

$$f_\theta(X) = h_\theta(X, A^{(1)}, \ldots, A^{(K)}) \equiv h_\theta(X, \mathcal{H}). \tag{8}$$

In what follows, we characterize the condition for correct equivariance (where h_θ is equivariant to G but not $\mathbf{G} \backslash G$) when h_θ is a one-layer message-passing neural network (MPNN) operating on nodes and edges. This argument immediate extends to h_θ being a hypergraph network operating on hyper-edges with deeper layers. By the definition of message-passing, the output h_θ at node i is computed as

$$h_\theta(X, A^{(2)})[i] = \phi\left(\psi_n(X_i), \tau\left(\{\!\{\psi_e(X_i, X_j, A^{(2)}_{i,j}) \mid j \in \mathcal{N}_A(i)\}\!\}\right)\right) \tag{9}$$

where ψ_e is the edge update function, ψ_n, ϕ are the node update functions, and τ is the edge multiset aggregation. We prove the following result:

Lemma 2 *If h_θ uses injective functions for (hyper-)edge feature update ψ_e, node update ϕ, and (hyper-)edge multiset aggregation τ, and if the node features are distinct, then h_θ is not equivariant to permutations in $S_n \setminus G$.*

Hypergraph Construction and Approximation with 2-Closure Achieving exact symmetry breaking of S_n to the desired subgroup G may require a hypergraph $\mathcal{H}$ of prohibitively high order (up to $K \leq n$). For efficiency, we fix $K = 2$ and construct positional and edge features $\mathcal{H} = (A^{(1)}, A^{(2)})$, whose automorphism group $\mathrm{Aut}(\mathcal{H})$ reflects on how G acts on nodes and pairs of nodes. Concretely, nodes i, j (or node pairs $(i_1, i_2), (j_1, j_2)$) are assigned the same feature if and only if they are in the same G-orbit. In this way, the positional and edge features encode the orbit partition under G. By construction, $\mathrm{Aut}(A^{(2)})$ is the 2-closure group of G, denoted as $G^{(2)}$ [23]. See Appendix B for the concrete algorithm and details. In general $G \leq G^{(2)}$, and in many cases $G = G^{(2)}$ regardless of the permutation representations; such groups are called *totally 2-closed*. This class includes finite nilpotent groups that are either cyclic or a direct product of a generalized quaternion group with a cyclic group of odd order [2]. See Figure 1 for illustrative examples.

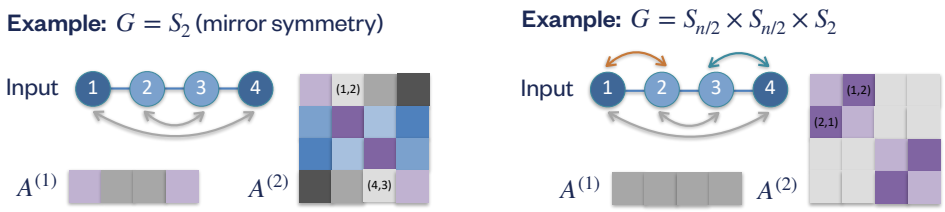


Figure 1: Example symmetry breaking objects as positional features $A^{(1)}$ and edge features $A^{(2)}$ for encoding subgroup symmetries in 4-node paths. These symmetries are explored further in Section 3 (i.e., sequence tasks described in Table 1). Note that these symmetry groups are equal to their 2-closures.

3. Experimental Results

Towards developing an equivariant foundation model, we evaluate ASEN in diverse settings by answering the following questions: (1) Can ASEN — with a single architecture — match the performance of prior works that use distinct models for each symmetry or non-equivariant baselines? (2) Can ASEN leverage shared symmetry structure across tasks to outperform task-specific equivariant models in multitask learning and transfer learning?

Single Architecture Performance ASEN with a single architecture achieves competitive performance for graph learning applications (Appendix D.1) compared to prior work [14] that requires multiple distinct equivariant models, highlighting its flexibility in implementing diverse symmetry groups. As shown in Figure 2, ASEN with the task-specific symmetry notably outperforms its non-equivariant baseline across all three sequence tasks (Table 1).

Learning Shared Symmetry Figure 3 shows that pretraining ASEN achieves notably better performance compared to training from scratch, showcasing the ability of ASEN to transfer structural knowledge. Figure 5 shows that in multitask learning, ASEN can leverage similar symmetries across tasks for faster optimization and improved generalization.

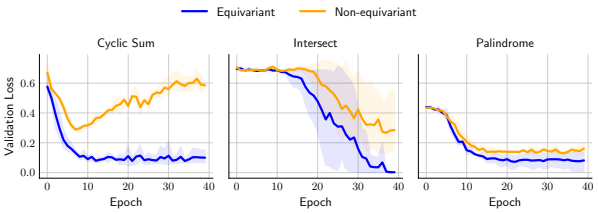


Figure 2: ASEN with the correct group ("Equivariant") converges faster and to a lower loss across tasks than its trivial symmetry counterpart ("Non-equivariant").

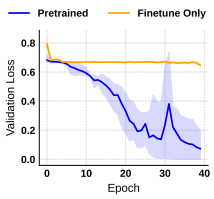


Figure 3: Equivariant transfer learning on Task Intersect: faster convergence and lower loss.

Acknowledgments

The authors thank Erik Thiede (Cornell), Soledad Villar (JHU), Bruno Ribeiro (Purdue), and Risi Kondor (UChicago) for valuable discussions. The authors also thank the anonymous reviewers and area chair for their constructive feedback. H.L. is supported by the Fannie and John Hertz Foundation.

References

[1] Ralph Abboud, Ismail Ilkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. 2021.

[2] Alireza Abdollahi and Majid Arezoomand. Finite nilpotent groups that coincide with their 2-closures in all of their faithful permutation representations. *Journal of Algebra and Its Applications*, 17(04):1850065, 2018.

[3] Matthew Ashman, Cristiana Diaconu, Adrian Weller, Wessel Bruinsma, and Richard E Turner. Approximately equivariant neural processes. *arXiv preprint arXiv:2406.13488*, 2024.

[4] Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew G Wilson. Learning invariances in neural networks from training data. *Advances in neural information processing systems*, 33:17605–17616, 2020.

[5] Beatrice Bevilacqua, Joshua Robinson, Jure Leskovec, and Bruno Ribeiro. Holographic node representations: Pre-training task-agnostic node embeddings. In *The Thirteenth International Conference on Learning Representations*, 2025.

[6] Alberto Bietti, Luca Venturi, and Joan Bruna. On the sample complexity of learning under geometric stability. *Advances in neural information processing systems*, 34: 18673–18684, 2021.

[7] Ben Blum-Smith, Ningyuan Huang, Marco Cuturi, and Soledad Villar. Learning functions on symmetric matrices and point clouds via lightweight invariant features, 2024. URL https://arxiv.org/abs/2405.08097.

[8] Alexander Bogatskiy, Timothy Hoffman, David W Miller, Jan T Offermann, and Xiaoyang Liu. Explainable equivariant neural networks for particle physics: Pelican. *Journal of High Energy Physics*, 2024(3):1–66, 2024.

[9] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[10] Bryn Elesedy and Sheheryar Zaidi. Provably strict generalisation benefit for equivariant models. In *International conference on machine learning*, pages 2959–2969. PMLR, 2021.

[11] Marc Finzi, Gregory Benton, and Andrew G Wilson. Residual pathway priors for soft equivariance constraints. *Advances in Neural Information Processing Systems*, 34: 30037–30049, 2021.

[12] Marc Finzi, Max Welling, and Andrew Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In *International conference on machine learning*, pages 3318–3328. PMLR, 2021.

[13] Sharut Gupta, Chenyu Wang, Yifei Wang, Tommi Jaakkola, and Stefanie Jegelka. In-context symmetries: Self-supervised learning through contextual world models. *Advances in Neural Information Processing Systems*, 37:104250–104280, 2024.

[14] Ningyuan Huang, Ron Levie, and Soledad Villar. Approximately equivariant graph networks. *arXiv preprint arXiv:2308.10436*, 2023.

[15] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7): 1325–1339, 2014. doi: 10.1109/TPAMI.2013.248.

[16] Jan W Jaworowski. Extensions of g-maps and euclidean g-retracts. *Mathematische Zeitschrift*, 146(2):143–148, 1976.

[17] Hyunsu Kim, Hyungi Lee, Hongseok Yang, and Juho Lee. Regularizing towards soft equivariance under mixed symmetries. In *International Conference on Machine Learning*, pages 16712–16727. PMLR, 2023.

[18] Richard Lashof. The equivariant extension theorem. *Proceedings of the American Mathematical Society*, 83(1):138–140, 1981.

[19] Hannah Lawrence, Vasco Portilheiro, Yan Zhang, and Sékou-Oumar Kaba. Improving equivariant networks with probabilistic symmetry breaking. In *ICML 2024 Workshop on Geometry-grounded Representation Learning and Generative Modeling*, 2024.

[20] Derek Lim, Haggai Maron, Marc T. Law, Jonathan Lorraine, and James Lucas. Graph metanetworks for processing diverse neural architectures. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ijK5hyxs0n.

[21] Haggai Maron, Ethan Fetaya, Nimrod Segol, and Yaron Lipman. On the universality of invariant networks. In *International conference on machine learning*, pages 4363–4371. PMLR, 2019.

[22] Mircea Petrache and Shubhendu Trivedi. Approximation-generalization trade-offs under (approximate) group equivariance. *Advances in Neural Information Processing Systems*, 36:61936–61959, 2023.

[23] Ilia Ponomarenko and Andrey Vasil'ev. Two-closures of supersolvable permutation groups in polynomial time. *computational complexity*, 29(1):5, 2020.

[24] Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Equivariance through parameter-sharing. In *International conference on machine learning*, pages 2892–2901. PMLR, 2017.

[25] Ryoma Sato, Makoto Yamada, and Hisashi Kashima. Random features strengthen graph neural networks. In *Proceedings of the 2021 SIAM international conference on data mining (SDM)*, pages 333–341. SIAM, 2021.

[26] Tess E Smidt, Mario Geiger, and Benjamin Kurt Miller. Finding symmetry breaking order parameters with euclidean neural networks. *Physical Review Research*, 3(1): L012002, 2021.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[28] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *6th International Conference on Learning Representations*, 2017.

[29] Rui Wang, Robin Walters, and Rose Yu. Approximately equivariant networks for imperfectly symmetric dynamics. In *International Conference on Machine Learning*, pages 23078–23091. PMLR, 2022.

[30] Helmut W. Wielandt. Permutation groups through invariant relations and invariant functions. 1969. URL https://api.semanticscholar.org/CorpusID:117295077.

[31] YuQing Xie and Tess Smidt. Equivariant symmetry breaking sets. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=tHKH4DNSR5.

Appendix A. Related Work

Subgroup Equivariance and Symmetry Breaking In equivariant network design, recent works [7; 3; 20] have proposed subgroup-equivariant models via augmenting an auxiliary input. Specifically, Blum-Smith et al. [7] proposed a permutation-invariant model for symmetric matrices by using a DeepSet base model—invariant to a bigger group, together with a suitable symmetry-breaking parameter to reduce the base model symmetries; Ashman et al. [3] used fixed symmetry breaking inputs to construct non-equivariant models or approximately equivariant ones. Symmetry breaking via node identification is a popurlar technique to enhance the expressivity of graph neural networks [1; 25; 5]. Symmetry breaking of the *input* has also been used to improve the flexibility of equivariant models for applications in graph generation and physical modeling [26; 19; 31]. Unlike [26; 19; 31] that perform input-dependent symmetry breaking, we break the symmetry of the *model* uniformly for all input. Moreover, existing works typically focus on (approximate) equivariance to one particular group. In contrast, we build a single model capable of modeling diverse data equivariantly, via different choices of $\mathbf{v}$ adapted to the target application.

Approximate and Adaptive Equivariance Another direction towards flexible equivariant networks relies on approximate equivariance [29; 14], soft equivariance by converting architectural constraints into a prior [4; 11], regularization [17], or adaptive equivariance per task and environment [13]. Approximate equivariance can also improve generalization [29; 14]. This motivates our approach that approximates the automorphism group of the symmetry breaking input with the 2-closure group.

Appendix B. Details of Section 2 Hypergraph Approximation with 2-Closure

Our procedure is summarized in Algorithm 1, with additional details follow thereafter.

Algorithm 1 Compute Edge Orbits $A^{(2)}$ such that $\mathrm{Aut}(A^{(2)}) = G^{(2)}$ (SymPy commands)

Require: Generators $\sigma_1, \dots, \sigma_r$ of $G \leq S_n$

Ensure: Edge orbits $A^{(2)} \in [n] \times [n]$ where $A^{(2)}_{ij} = A^{(2)}_{mn} \iff (i,j) \sim_G (m,n)$.

1: **Lift generators:** For each $\sigma_i \in S_n$, define $\rho_i \in S_{n^2} : (x_a, x_b) \mapsto (\sigma_i(x_a), \sigma_i(x_b))$.
Encode (x_a, x_b) as $a \cdot n + b$ and construct ρ_i as `Permutation` of size n^2.

2: **Form diagonal subgroup:** Let $\Delta(G) \coloneqq \langle \rho_1, \dots, \rho_r \rangle$ be the subgroup of S_{n^2}.
`Delta = PermutationGroup([`$\rho_1, \dots, \rho_r$`])`.

3: **Compute edge orbits:** For an edge (x_a, x_b), apply ρ_i repeatedly until no new pairs can be found.
`Delta.orbits()`

1. **Lift the generators.** Given the generators $\sigma_1, \dots, \sigma_r$ of $G \leq S_n$, we define new permutations ρ_i acting on $X \times X$ with $X = \{1, \dots, n\}$. Each ρ_i acts diagonally:

$$\rho_i \colon (x_a, x_b) \;\mapsto\; (\sigma_i(x_a), \sigma_i(x_b)).$$

In practice (e.g. in `sympy`), if n is the degree of the action, we encode (x_a, x_b) as a single index $a \cdot n + b$, and construct ρ_i as a `Permutation` of size n^2.

2. **Form the diagonal subgroup.** Define
$$\Delta(G) \;=\; \langle \rho_1, \ldots, \rho_r \rangle,$$
the subgroup of S_{n^2} generated by the lifted permutations.

   Conceptually, we take the subgroup generated by the ρ_i. In `sympy`, this is done by calling
$$\texttt{Delta = PermutationGroup([}\rho_1, \ldots, \rho_r\texttt{])}.$$
   Internally, `PermutationGroup` builds a *base and strong generating set* (BSGS) for $\Delta(G)$ via Schreier–Sims. The BSGS consists of a chosen base $B = (b_1, \ldots, b_k)$ and *strong generators* adapted to the chain of stabilizers
$$\Delta(G) = G^{(0)} \geq G^{(1)} \geq \cdots \geq G^{(k)} = \{e\},$$
   where $G^{(i)}$ is subgroup fixing the first i base points. The associated *basic orbits* and *Schreier vectors* allow efficient navigation in the group.

3. **Compute the orbits.** The G–orbits on $X \times X$ are exactly the orbits of $\Delta(G)$. Concretely, starting from a pair (x_a, x_b), we apply the generators ρ_i repeatedly until no new pairs are found; the set obtained is its orbit. With a BSGS, orbit computation is polynomial-time as SymPy performs a breadth-first search on the strong generators and stores transversal information for reconstruction. In code, this is as simple as
```
Delta.orbits()
```
   which returns all $\Delta(G)$-orbits of the action.

Appendix C. Approximation and Universality Properties of ASEN

C.1. Proofs of Proposition 1 and Lemma 2

Proposition 1 *Let $h_\theta : \mathcal{X} \times \mathcal{V} \to \mathcal{Y}$ be $\mathbf{G}$-equivariant, and let $\mathrm{Aut}(\mathbf{v}) = G$. Then $f_\theta(x) := h_\theta(x, \mathbf{v})$ is equivariant to G. If additionally h_θ is injective in the input $\mathbf{v}$, then f_θ is not equivariant to any transformation in $\mathbf{G} \setminus G$.*

Proof We have already shown in Section 2 that f_θ is G-equivariant. Now, suppose that h_θ is injective in $\mathbf{v}$, and let $g \in \mathbf{G} \setminus G$. We want to show that $f_\theta(gx) \neq g f_\theta(x)$ for some choice of x. We choose any $x \in \mathcal{X}$. This holds because $\mathbf{v} \neq g\mathbf{v}$ (since otherwise $g \notin G = \mathrm{Aut}(\mathbf{v})$). Thus, by injectivity, $h_\theta(gx, \mathbf{v}) \neq h_\theta(gx, g\mathbf{v})$. This allows us to conclude that

$$\begin{aligned} f_\theta(gx) &= h_\theta(gx, \mathbf{v}) && (10)\\ &\neq h_\theta(gx, g\mathbf{v}) && (11)\\ &= g h_\theta(x, \mathbf{v}) && (12)\\ &= g f_\theta(x). && (13) \end{aligned}$$

This concludes the proof. ■

We remark that the injectivity of h_θ in the input $\mathbf{v}$ is sufficient to prove Proposition 1, but not necessary. As shown in the proof, this injectivity assumption allows us to show $f_\theta(gx) \neq g f_\theta(x)$ for *any* $x \in \mathcal{X}$, while establishing this for a particular choice of x suffices.

Lemma 2 *If h_θ uses injective functions for (hyper-)edge feature update ψ_e, node update ϕ, and (hyper-)edge multiset aggregation τ, and if the node features are distinct, then h_θ is not equivariant to permutations in $S_n \setminus G$.*

Proof We will show that for any $\sigma \in S_n \setminus G$, $f_\theta(\sigma X) \neq \sigma f_\theta(X)$. By (9), it suffices to show that

$$h_\theta(\sigma X, A^{(2)}) \neq h_\theta(\sigma X, \sigma A^{(2)}). \tag{14}$$

Let $\mathcal{N}_A(i) \coloneqq \{j \in [n] : A_{ij} \neq 0\}$ be the neighborhood of node i. Since $\sigma \in S_n \setminus G$, there exists a node i such that its original neighborhood differs from the permuted one, $\mathcal{N}_A(i) \neq \mathcal{N}_{\sigma A}(i)$. By definition of MPNN (9),

$$h_\theta(\sigma X, A^{(2)})[i] = \phi\left(\psi_n(X_\sigma(i)), \tau\left(\{\!\{\psi_e(X_\sigma(i), X_\sigma(j), A^{(2)}_{i,j}) \mid j \in \mathcal{N}_A(i)\}\!\}\right)\right) \tag{15}$$

$$h_\theta(\sigma X, \sigma A^{(2)})[i] = \phi\left(\psi_n(X_\sigma(i)), \tau\left(\{\!\{\psi_e(X_\sigma(i), X_\sigma(j), A^{(2)}_{\sigma(i),\sigma(j)}) \mid j \in \mathcal{N}_{\sigma A}(i)\}\!\}\right)\right). \tag{16}$$

Since $\mathcal{N}_A(i) \neq \mathcal{N}_{\sigma A}(i)$ and the assumption that $\{X_\sigma(j)\}_{j=1}^n$ has distinct elements, the multisets

$$\{\!\{(X_\sigma(i), X_\sigma(j), A^{(2)}_{i,j}) \mid j \in \mathcal{N}_A(i)\}\!\} \neq \{\!\{(X_\sigma(i), X_\sigma(j), A^{(2)}_{\sigma(i),\sigma(j)}) \mid j \in \mathcal{N}_{\sigma A}(i)\}\!\}.$$

By injectivity of ψ_e,

$$\{\!\{\psi_e(X_\sigma(i), X_\sigma(j), A^{(2)}_{i,j}) \mid j \in \mathcal{N}_A(i)\}\!\} \neq \{\!\{\psi_e(X_\sigma(i), X_\sigma(j), A^{(2)}_{\sigma(i),\sigma(j)}) \mid j \in \mathcal{N}_{\sigma A}(i)\}\!\}.$$

By injectivity of τ and ϕ, we have $h_\theta(\sigma X, A^{(2)})[i] \neq h_\theta(\sigma X, \sigma A^{(2)})[i]$, and thus $h_\theta(\sigma X, A^{(2)}) \neq h_\theta(\sigma X, \sigma A^{(2)})$. ■

C.2. Connections to Equivariant MLPs

Here, we show that ASEN can simulate certain configurations of a common type of equivariant neural network, sometimes called an *equivariant MLP*, which consists of equivariant linear maps and elementwise nonlinearities [24; 21; 12]. When applied to a group G that acts as permutation matrices on $\mathbb{R}^n$, an equivariant MLP is defined as a composition: $T_L \circ \sigma \circ \cdots \circ \sigma \circ T_1$, where σ is an elementwise nonlinearity, and each $T_i : \mathbb{R}^{n^{k_i}} \to \mathbb{R}^{n^{k_{i+1}}}$ is a G-equivariant linear map (for simplicity, we ignore channel dimension here). We call $k^* = \max_i k_i$ the *order* of the G-MLP, so if $T_i : \mathbb{R}^n \to \mathbb{R}^n$ for each i then the G-MLP has order 1. We prove the following result:

Theorem 3 *Any order 1 G-MLP can be approximated to arbitrary accuracy on a compact domain via ASEN with $K = 2$ and the two-closure approximation* $\mathrm{Aut}(A^{(2)}) = G^{(2)}$.

Proof We show that one layer of ASEN using a message-passing GNN backbone can simulate $\sigma \circ T_i$, so suppose $L = 1$ (i.e. the G-MLP has one layer). Recall that any equivariant linear map $T : \mathbb{R}^n \to \mathbb{R}^n$ can be viewed as a linear combination $T = \sum_{l=1}^{d} a_l B^l$, where $a_l \in \mathbb{R}$ are scalars, the $B^l : \mathbb{R}^n \to \mathbb{R}^n$ are G-equivariant linear maps that span the vector space of G-equivariant linear maps, and d is the dimension of this vector space of G-equivariant linear maps. Moreover, by an argument similar to Maron et al. 2019 "On the Universality of Invariant Networks" [21], we can define the B^l as follows:

Let $\tau_1, \ldots, \tau_q$ be the unique orbits of the action of G on node-pair indices $[n] \times [n]$ (we refer to these as node-pair orbits). Then Ravanbakhsh et al. [24]; Maron et al. [21] show that $q = d$, and the B^l can be chosen as:

$$B^l_{ij} = \begin{cases} 1 & (i,j) \in \tau_l \\ 0 & \text{else.} \end{cases} \tag{17}$$

This is a form of weight sharing, where B^l is constant on each node-pair orbit (and hence any T that is a linear combination of them is constant on each node-pair orbit). Note that the linear map can be shown to take a message-passing form as follows. For an input $x \in \mathbb{R}^n$, let x_i be viewed as a node representation for node i. Then the new node representation for node i after this layer is

$$T(x)_i = \sum_{l=1}^{q} a_l (B^l x)_i \tag{18}$$

$$= \sum_{l=1}^{q} a_l \sum_{j=1}^{n} B^l_{ij} x_j \tag{19}$$

$$= \sum_{j=1}^{n} x_j \sum_{l=1}^{q} a_l B^l_{ij} \tag{20}$$

This can be interpreted as message passing, where the node j passes message $x_j \sum_{l=1}^{q} a_l B^l_{ij}$ to the node i. We will show that ASEN with order $K = 2$ and two-closure automorphism group approximation can simulate this map $T(x)$. Note that any $H \in \mathbb{R}^{n^2}$ that has $\mathrm{Aut}(H) = G^{(2)}$ must satisfy that $H_{i_1,j_1} = H_{i_2,j_2}$ if and only if $(i_1, j_1) \sim_G (i_2, j_2)$. In other words, $H_{i_1,j_1} = H_{i_2,j_2}$ if and only if (i_1, j_1) and (i_2, j_2) are in the orbit τ_l for some l. Let the distinct entries of H be denoted by $h_l \in \mathbb{R}$, so that $h_l = H_{i_1,j_1}$ if and only if $(i_1, j_1) \in \tau_l$.

Finally, we define the GNN h_θ to take the following message-passing-based form. Let x_i be the representation of node i. The GNN updates the node representations to $\hat{x}_i$ via two

multilayer perceptrons $\text{MLP}^e(h) : \mathbb{R} \to \mathbb{R}^q$ and $\text{MLP}^v : \mathbb{R}^{q+1} \to \mathbb{R}$ as follows:

$$\hat{x}_i = \sum_{j=1}^{n} \text{MLP}^v(x_j, \text{MLP}^e(H_{i,j})) \tag{21}$$

$$\text{MLP}^{\text{e}}(h)_l = \begin{cases} 1 & \text{if } h = h_l \\ 0 & \text{else} \end{cases} \tag{22}$$

$$\text{MLP}^v(x, y) = x \sum_{l=1}^{q} a_l y_l. \tag{23}$$

Note that $\text{MLP}^e(H_{i,j}) = B^l_{ij}$ so that $\text{MLP}^v(x_j, \text{MLP}^e(H_{i,j})) = x_j \sum_{l=1}^{q} a_l B^l_{ij}$, which shows that $\hat{x}_i = T(x)_i$. In practice, an MLP cannot exactly express these functions, but an MLP can approximate each function to arbitrary precision $\epsilon > 0$ on a compact domain. Note that the function MLP^e seems discontinuous, but it is only defined on finitely many inputs, so it has a continuous extension that is exact on the finite inputs. ■

We remark that much like an equivariant MLP can increase expressivity by increasing the order k^*, we can increase expressivity in ASEN by increasing $K \geq 2$ and using the K-closure group approximation [23]. While we show this relationship in expressivity at $k^* = 1$ and $K = 2$, we believe that there may be relationships between the two methods at higher orders, and leave it to future work.

C.3. Universality Results

We next show that the universality of ASEN follows from the universality of its base model.

Theorem 4 *Let $\mathbf{G}$ be a compact group, $\mathcal{X}, \mathcal{V}$ be compact metric $\mathbf{G}$-spaces, and $\mathcal{Y}$ be a compact $\mathbf{G}$-space. Let $f_\theta : \mathcal{X} \times \mathcal{V} \to \mathcal{Y}$ be a universal family of continuous $\mathbf{G}$-equivariant networks, i.e. $f_\theta(gx, gv) = g \cdot f_\theta(x, v)$. Consider $\mathcal{H} \in \mathcal{V}$ with stabilizer equal to a subgroup $G \leq \mathbf{G}$. Then, the family $f_\theta(\cdot, \mathcal{H})$ is universal over continuous G-equivariant functions from $\mathcal{X}$ to $\mathcal{Y}$.*

Proof Let $f^* : \mathcal{X} \to \mathcal{Y}$ be a continuous, G-equivariant function. To prove universality of $f_\theta(\cdot, \mathcal{H})$, we must show that for any ϵ, there exists a θ such that $f_\theta(\cdot, \mathcal{H})$ is ϵ-close to f^*. To achieve this, let's first define a new function (which we will prove is $\mathbf{G}$-equivariant), $n : \mathcal{X} \times \mathcal{O} \to \mathcal{Y}$ where $\mathcal{O} = \{g\mathcal{H} \ : g \in \mathbf{G}\}$ as follows: for any $g \in \mathbf{G}$,

$$n(x, g\mathcal{H}) \coloneqq g f^*(g^{-1}x). \tag{24}$$

Note in particular that $n(x, \mathcal{H}) = f^*(x)$. To first show that is a valid definition of n, we will argue that if $g\mathcal{H} = g'\mathcal{H}$, then $gf^*(g^{-1}x) = g'f^*(g'^{-1}x)$ for all $x \in \mathcal{X}$. By the assumption $g\mathcal{H} = g'\mathcal{H}$, we have $g' = gs$ for some $s \in G$, the stabilizer of $\mathcal{H}$. Then

$$g'f^*(g'^{-1}x) = (gs)f^*((gs)^{-1}x) = g(sf^*(s^{-1}g^{-1}x)) = gf^*(g^{-1}x), \tag{25}$$

where the last equality follows from f^* being G-equivariant.

Next, the continuity of n on $\mathcal{X} \times \mathcal{O}$ follows from the continuity of f^*, the inversion map g^{-1} and group action g.

Finally, to show n is $\mathbf{G}$-equivariant, we note that for any $h \in \mathbf{G}$,

$$n(hx, hg\mathcal{H}) = hgf^*\left((hg)^{-1}hx\right) = hgf^*(g^{-1}x) = hn(x, g\mathcal{H}). \tag{26}$$

Thus, n is a continuous $\mathbf{G}$-equivariant function on $\mathcal{X} \times \mathcal{O}$. By universality of $\{f_\theta\}$ on $\mathcal{X} \times \mathcal{V} \mapsto \mathcal{Y}$ (under the supremum norm) and the orbit $\mathcal{O}$ being closed in $\mathcal{V}$, $\{f_\theta\}$ is also universal on the subdomain $\mathcal{X} \times \mathcal{O}$. In particular, for any ϵ, there exists some θ such that f_θ approximates n ϵ-well over all of $\mathcal{X} \times \mathcal{O}$. Thus, it must also approximate n ϵ-well over $\mathcal{X} \times \{\mathcal{H}\}$. But since $n = f^*$ on $\mathcal{X} \times \{\mathcal{H}\}$, this completes the proof. ■

We remark that Theorem 4 can be generalized to locally compact groups, via Jaworowski's equivariant extension theorem [16; 18].

Appendix D. Details of Section 3 Experiments

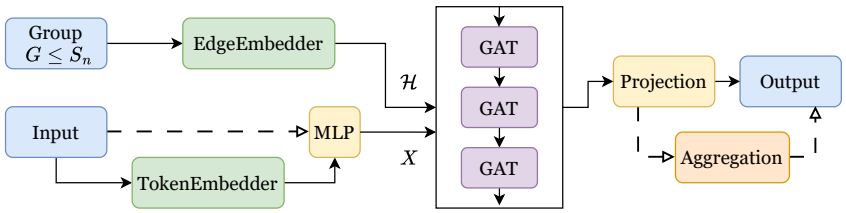


Figure 4: ASEN Architecture to model any permutation subgroup-equivariant functions (dashed lines represent the invariant model)

Architecture Our backbone is a permutation-invariant graph neural network (GNN), composed of input layers (e.g. embedding, MLP), followed by four layers of GATv2 message-passing [28], and concluded with output layers (e.g., projection, aggregation); see Figure 4. Standard dropout and layer normalization are applied throughout. There are two task-specific modules: *EdgeEmbedder* that calls Algorithm 1 to categorize edge orbits and learn their embeddings; *TokenEmbedder* that maps (discrete) node features to token embeddings for classification tasks (omitted for regression tasks). While ASEN models equivariant functions in general, it can learn invariant functions by adding an invariant aggregation layer at the end of the architecture (for more efficient training, we disable the TokenEmbedder for the invariant setting). As *EdgeEmbedder* is a learnable module, ASEN can *discover more symmetries* from data if the chosen group $G^{(2)}$ (specifying the edge orbits) is smaller than the target group.

Table 1: Description of synthetic tasks, objective, and their corresponding symmetry group.

Synthetic Tasks		
Task	Objective	Symmetry
Intersect	Given two sequences of length $\frac{n}{2}$, determine which elements of each sequence are present in the other	$S_{n/2} \times S_{n/2} \times S_2$: sequences can be reordered, and the two sequences can be swapped
Palindrome	Given a sequence of length n, determine where the sequence has a contiguous subsequence that is a palindrome of length k, if one exists	Sequence reversal
Cyclic Sum	Given a sequence of length n, determine which cyclic contiguous subsequence of length k has the largest sum	C_n (cyclic shifts)

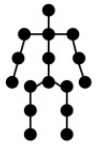
Experiment Details for Sequence Modelling Tasks (Figure 2) The dataset per task contains 2500 examples of sequence length 10, chosen to highlight the benefits of symmetry in data-scarce regimes. We train ASEN in the binary node classification setting to learn an equivariant mapping from the input node features to output node labels, both input and output represented as sequences. Our model is optimized using standard cross-entropy loss, with hidden dimension of 128, batch size of 64, learning rate of 0.01, and run on 40 epochs.

Experiment Details for Transfer Learning (Figure 3) We focus on the same set of synthetic tasks as Figure 2 (i.e., Cyclic Sum, Intersect, Palindrome) and simulate a low-resource regime by limiting access to only 0.15 units (375 datapoints) of training data. We compare two ASEN model variants: A *finetune-only* baseline trained from scratch using only the chosen task data, and a *pretrained* model initialized via joint training on 1.5 units each of the other tasks, followed by fine-tuning on the chosen task data. To encourage knowledge retention, we reduce the learning rate of the GNN backbone during fine-tuning, while allowing the embedding layers to update more freely.

D.1. Experiments on Symmetry Selection

We apply ASEN to perform symmetry model selection for learning on a fixed graph setting, using the experimental set-up from [14]. Unlike [14] that requires distinct G-equivariant layers for each group, ASEN offers a unified architecture to flexibly model different G-equivariance by using a corresponding symmetry-breaking parameter $\mathcal{H}$.

Human Pose Estimation We begin with an application in human pose estimation, using the Human3.6M dataset [15], which consists of 3.6 million human poses from various images. Our input features consist of 2D coordinates $X \in \mathbb{R}^{16\times 2}$ representing joint positions on a skeleton graph $A \in \{0,1\}^{16\times 16}$ (see Figure inset). The model predicts the corresponding 3D joint positions in $\mathbb{R}^{16\times 3}$.



Performance is evaluated using the standard P-MPJPE (Procrustes-aligned Mean Per Joint Position Error) metric. We consider three edge frameworks $\mathcal{H} = A^{(2)}$ (Algorithm 1): (1) fully-connected $A_f^{(2)}$, (2) sparse $A_s^{(2)}$ where edges are constrained to the support of the skeleton graph A, and (3) weakly sparse combining $A_f^{(2)} + A_s^{(2)}$. We consider *selecting* different automorphism groups of the human skeleton edges $\mathrm{Aut}(A^{(2)})$ (that yields the best performance): S_2 (full left-right reflection), S_2^2 (left arm/right arm and left leg/right leg), S_2^6 (each left-side joint independently mapped to corresponding joint on right side), and I (no equivariance). Our results in Table 2 obtained from a single model ASEN match with those reported in [14] that require multiple distinct equivariant architectures. Notably, the weakly sparse graph yields some of the strongest results, highlighting ASEN's flexibility in capturing multiple symmetries. This approach can also be used to represent a graph with two sets of symmetry.

Table 2: P-MPJPE error (↓) for human pose estimation using different symmetry groups and edge frameworks.

Group	Fully Connected	Sparse	Weakly Sparse
I	34.71	**33.39**	34.75
S_2	39.48	40.52	**38.80**
S_2^2	43.24	42.37	**40.67**
S_2^6	47.54	49.45	**46.52**

Table 3: Mean Absolute Error (MAE ↓) for traffic flow prediction.

Model	MAE
Fully Connected, $S_{n_1} \cdot S_{n_2}$	2.72
Sparse, $S_{n_1} \cdot S_{n_2}$	**2.69**
Fully Connected, $S_{n_1} \cdot \cdots \cdot S_{n_9}$	2.79
Sparse, $S_{n_1} \cdot \cdots \cdot S_{n_9}$	2.77
DCRNN	2.77

Traffic Prediction Next, we evaluate on the traffic forecasting task, also taken from [14]. This uses the METR-LA dataset, containing time-series traffic data from 207 sensors deployed on Los Angeles highways. Each sensor node reports speed and volume measurements, yielding input features $X_t \in \mathbb{R}^{207\times 2}$ every 5 minutes. The objective is to predict traffic conditions at future time steps $X_{t+1}, X_{t+2}, X_{t+3}$ based on the historical sequence X_{t-2}, X_{t-1}, X_t. The underlying graph structure is defined via a sensor adjacency matrix $A \in \mathbb{R}^{207\times 207}$ constructed from roadway connectivity.

To incorporate symmetry structure, we leverage the spatial layout of sensors along major highways. We consider two group structures, taken from [14]: one with two symmetric clusters representing major highway branches $G = S_{n_1} \times S_{n_2}$, and another with nine clusters corresponding to finer-grained regional groupings $G = S_{n_1} \times \ldots \times S_{n_9}$. These symmetry groups serve as approximate equivariances, which encourage learning invariant representations for similarly situated sensors. We choose $\mathcal{H} = A^{(2)}$ such that $\mathrm{Aut}(A^{(2)}) = G$ via Algorithm 1, and benchmark our model in both fully-connected and sparse graph regimes. As shown in Table 3, ASEN with a unified model achieves competitive performance as [14] that requires distinct equivariant models for each symmetry group.

D.2. Multitask Learning Applications

In this section, we investigate whether learning related tasks with compatible or similar symmetry groups can benefit ASEN from shared representation learning. We again make use

of the synthetic tasks introduced in Table 1. While each task can be learned independently using a different equivariant model, we assess whether *multitask training* can facilitate improved generalization in low-data regimes. To this end, we use the same ASEN backbone (including weights) shared across tasks, while the TokenEmbedder and EdgeEmbedder modules are task-specific. During training, we randomly sample batches from all tasks, ensuring concurrent and balanced updates across tasks.

To focus on performance under constrained data availability, we limit the maximal training set size to 2,500 datapoints (referred to as "1 unit"). We then compare two regimes: one training only on r units of a *single-task*, and the other training on r units on all tasks, specifically Intersect, Cyclic Sum, and Palindrome. We vary $r \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ and report the average performance across three random seeds. Figure 5 shows that multitask training leads to significantly improved convergence and test accuracy in the low-data setting for learning Intersect, demonstrating the benefit of symmetry-aligned task transfer. Meanwhile, Table 4 shows that the multitask performance on the other two tasks (Cyclicsum, Palindrome) remains similar to the single task setting.

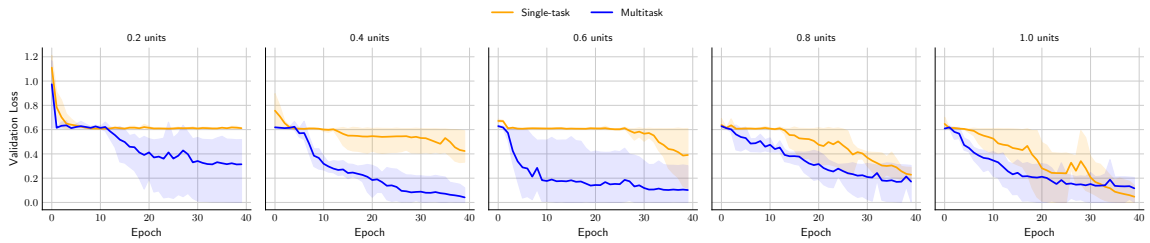


Figure 5: Multitask versus Single-task performance on Task Intersect with varying training set sizes

Table 4: Multi-task Test Losses (0.08 units)

Method	**Cyclicsum**	**Palindrome**
Single-task	0.3869	0.510
Multi-task	0.3839	0.537