Higher-Order Message Passing for Glycan Representation Learning

Roman Joeres
Department of Chemistry and Molecular Biology and
Wallenberg Centre for Molecular and Translational Medicine
University of Gothenburg
Gothenburg, Sweden
and
Saarland Informatics Campus
Saarland University
Saarbruecken, Germany

Daniel Bojar
Department of Chemistry and Molecular Biology and
Wallenberg Centre for Molecular and Translational Medicine
University of Gothenburg
Gothenburg, Sweden
daniel.bojar@gu.se

Abstract

Glycans are the most complex biological sequence, with monosaccharides forming extended, non-linear sequences. As post-translational modifications, they modulate protein structure, function, and interactions. Due to their diversity and complexity, predictive models of glycan properties and functions are still insufficient.

Graph Neural Networks (GNNs) are deep learning models designed to process and analyze graph-structured data. These architectures leverage the connectivity and relational information in graphs to learn effective representations of nodes, edges, and entire graphs. Iteratively aggregating information from neighboring nodes, GNNs capture complex patterns within graph data, making them particularly well-suited for tasks such as link prediction or graph classification across domains.

This work presents a new model architecture based on combinatorial complexes and higher-order message passing to extract features from glycan structures into a latent space representation. The architecture is evaluated on an improved GlycanML benchmark suite, establishing a new state-of-the-art performance. We envision that these improvements will spur further advances in computational glycosciences and reveal the roles of glycans in biology.

1 Introduction

Glycans are complex carbohydrate structures composed of covalently connected monosaccharides forming branching oligosaccharide trees [1]. Glycans play crucial roles in numerous biological processes, from regulating the immune response to mediating host-pathogen interactions [2, 3]. As post-translational modifications, glycans are also key for modulating protein structure, function, and interactions. Especially understanding glycan-protein interactions has recently shed light onto

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Dataset Name	Class.	# Mono per Glycan	# Atoms per Glycan	#train/#val/#test
Immunogenicity	Binary	6.30	97.54	825/230/113
Glycosylation	3-class	7.59	115.97	1,134/317/163
Tax. Domain	5-label	7.13	91.43	11,378/3,170/1,579
Tax. Kingdom	15-label	7.13	91.43	11,378/3,170/1,579
Tax. Phylum	43-label	7.13	91.43	11,378/3,170/1,579
Tax. Class	78-label	7.13	91.43	11,378/3,170/1,579
Tax. Order	170-label	7.13	91.43	11,378/3,170/1,579
Tax. Family	265-label	7.13	91.43	11,378/3,170/1,579
Tax. Genus	394-label	7.13	91.43	11,378/3,170/1,579
Tax. Species	569-label	7.13	91.43	11,378/3,170/1,579

Table 1: Benchmark task description.

receptors for SARS-CoV-2 [4], immune checkpoints in cancer [5], or erythrocyte invasion by malaria-causing parasites [6].

Traditional approaches to glycan analysis have relied on hand-crafted features or simple molecular descriptors, which often fail to capture the full structural complexity of these molecules. Recent advances in machine learning, particularly in graph neural networks (GNNs), have shown promise in addressing these limitations [7, 8]. GNNs can naturally represent the branched, non-linear structure of glycans and learn relevant features directly from unprocessed data.

Despite these advances, existing GNN models for glycan analysis often struggle to simultaneously capture both the atomic-level details and the higher-order structural information of glycans. Models focusing on atomic-level representations [9] may miss important topological features, while those operating on a coarser, monosaccharide level [8] may overlook crucial atomic interactions.

To address these challenges, we present GIFFLAR (Glycan Informed Foundational Framework for Learning Abstract Representations), a novel GNN architecture specifically designed for glycan representation learning. GIFFLAR leverages combinatorial complexes [10] to represent glycans at multiple levels of abstraction—atoms, bonds, and monosaccharides—within a single, unified framework. This multi-level representation, combined with higher-order message passing, enables the model to learn rich, hierarchical features that capture both local and global structural information.

We evaluate GIFFLAR on an expanded and curated version of the GlycanML benchmark suite [11], encompassing a diverse set of glycan property prediction tasks. Our experiments demonstrate that GIFFLAR consistently outperforms existing methods across all tasks, including traditional machine learning approaches and state-of-the-art GNN models.

The rest of this paper is organized as follows: Section 2 discusses related work in glycan analysis and graph neural networks. Section 3 describes our experimental setup, including data preparation and model architecture. Section 4 presents our evaluation results and ablation studies. Finally, Section 5 discusses our findings and outlines limitations and future work directions. The necessary background on combinatorial complexes and higher-order message passing can be found in Appendix A.

2 Related Work

GlycanML In 2024, Xu et al. proposed to use existing benchmark datasets curated by our group to assess the performance of models for glycan property prediction [11]. This data was mainly taken from the 2021 version of SugarBase [12], and GlyConnect [13] for the glycosylation classification. In 2024, glycowork v1.3 was released [14], and the included data extends the taxonomy datasets beyond the coverage of GlycanML, comprising 40% new glycans in our extended benchmark datasets here. For the presented work, we thus created a dataset comprising eight taxonomy classification tasks from glycowork and two classification tasks for glycosylation and immunogenicity from GlycanML. As our model requires atomic graphs, we filtered the ten datasets for those fully specified glycans that could be translated from IUPAC to SMILES using GlyLES [15].

This previous work showed that the RGCN (Relational Graph Convolutional Network) [16] was the best on the single-task datasets. Therefore, we implemented this architecture as one of the baselines and the only one using the same heterogeneous graph architecture our new model will use. All models in this study were retrained on the same dataset of glycans, using the same data split across all models.

SweetNet The standard model to predict properties of glycans is SweetNet, which operates on a topological level, i. e., the tree of monomers [8]. As one of the first glycan-focused geometric deep learning models, it uses a GNN architecture for feature extraction and a simple MLP as the prediction head. Burkholz et al. showed the superiority of their approach over simple baselines. In downstream experiments, they showed that SweetNet can identify/extract essential features of glycans. In separate work, Lundstrøm et al. used SweetNet as an encoder for glycans in a lectin-glycan interaction prediction model [17] and Kellman et al. used it as an encoder to predict glycosylation potential at a protein glycosite [18].

GNNGLY After GlyLES was published, Alkuhlani et al. presented GNNGLY, a GCN-based model for glycan property prediction based on atomic structures computed with GlyLES [9]. This model was trained on the original taxonomy classification data from SugarBase. They showed improvement over simple baselines and comparable results to SweetNet. However, the study raises several concerns as it is not reproducible in our hands, and SweetNet was not retrained for the initial study. The latter is necessary for comparability, as the initial dataset for training SweetNet and GNNGLY was the same. Due to structural ambiguities, GlyLES could not convert all IUPAC-condensed strings of this dataset to SMILES, and the usable data for training and evaluating GNNGLY is different from the one used initially for SweetNet. Due to the lack of code and the insufficient architecture description for GNNGLY, we had to implement the model as best as possible.

GLAMOUR To our knowledge, GLAMOUR is the only published model incorporating atomic and macromolecular (topological) structures of molecules into one model [19]. Since it applies to any class of multimeric molecules, it has been tested for glycan property prediction. GLAMOUR operates on a topological level like SweetNet, with nodes representing monomeric units and edges representing their covalent connections. The atomic structures of the monomers and their connections are used to compute fingerprints as the initial features of the nodes and edges. GLAMOUR offers five GNN architectures as the backend of the model. However, we could not apply GAT and GCN to monosaccharides. Among the other three, namely MPNN, Weave, and AttentiveFP, the MPNN performed best (see Appendix B). Therefore, we used this as the baseline.

3 Experiments

3.1 Data Preparation

In total, we trained nine models on four representations of glycans. Simple baselines, namely Random Forests [20], Support Vector Machines [21], Gradient Boosting [22], and Multilayer Perceptrons [23], on atom-level Morgan Fingerprints, GNNGLY on atom-level homogeneous graphs, SweetNet and GLAMOUR on topology-level homogeneous graphs, and RGCN and GIFFLAR on heterogeneous graphs. Because all models except SweetNet require atomic graphs as input, we translated all IUPAC-condensed notations to SMILES using GlyLES [15] and filtered for those glycans for which GlyLES could compute SMILES strings. We then computed atomic graphs from these SMILES strings using RDKit. This ensured that all models were trained (see Appendix D) and evaluated using the same data. The data was split randomly, and all models were trained on the same split.

Similar to GlycanML, we trained the Immunology and Glycosylation datasets as binary and three-class, single-label classification tasks, respectively. In Alkuhlani et al. and Xu et al. [9, 11], training taxonomy classification as a multi-class, single-label problem led to accuracy measurements that are hard to interpret because a perfect model would not achieve an accuracy of 1, since the same glycan may be conserved across multiple taxonomic groups. Therefore, we changed the task for taxonomy predictions to a multi-label classification. This also reduced the dataset sizes, as each glycan was now only present once with all its labels instead of once per label.

The heterogeneous graphs for RGCN and GIFFLAR have three node types: atoms, bonds, and monosaccharides, corresponding to 0-cells, 1-cells, and 2-cells, respectively, as detailed in Appendix A. They are connected based on neighborhoods as defined in Equations 5 and 7 (in Appendix A), for intra-rank and inter-rank neighborhoods, respectively.

3.2 Model Architecture

For GIFFLAR, we used an architecture similar to Graph Isomorphism Networks [24] and instantiated equation 9 as

$$\mathbf{h}_x^{l+1} = \sum_{\mathcal{N}_k \in \mathcal{N}} \theta_k^l \left((1+\epsilon) \cdot \mathbf{h}_x^l + \sum_{y \in \mathcal{N}_k} \mathbf{h}_y^l \right), \text{ with} \tag{1}$$

$$\mathcal{N} = \{\mathcal{B}_{0,0},\ \mathcal{B}_{1,1},\ \mathcal{B}_{2,2},\ \mathcal{N}_{0,1}^{\downarrow},\ \mathcal{N}_{1,2}^{\downarrow}\} \tag{2}$$

and θ_k^l consisting of a fully connected layer, a parameterized ReLU activation function, dropout with $p = 0.2$, and batch normalization. The modules share the architecture across l and k but not the weights. The readout for the final graph embedding is a simple mean over all nodes. We explored other approaches, which did not result in higher performance (see Appendix C). The classification head is a simple, two-layered feed-forward network (FFN) with a parameterized ReLU activation function and a dropout layer ($p = 0.2$). The final model takes 128-dimensional node features as input. We use fixed, random embeddings per node class, e.g., C or O for atom nodes. The feature vectors are scaled up to 1024-dimensional embeddings in the hidden layers. As a result, the model has 35.1M trainable parameters.

All reported performance in this work is computed on the validation set. These performances were used to compare models and justify architectural decisions. Every model is biased toward the data that influenced its development [25]. Therefore, we use a separate holdout dataset to report the state-of-the-art performance on unseen data for the best model chosen on the validation set.

4 Evaluation

Combinatorial complexes present a natural choice to represent glycans as graph objects, combining their atomic representation and topological structure. We evaluated the performance of the models using accuracy, Area Under the Receiver Operating Characteristics (AUROC) curve, and Matthews Correlation Coefficient on all ten datasets. The three metrics cannot immediately be aggregated and compared, as a gain of .1 in accuracy differs from gaining .1 in AUROC and this varies between datasets. Therefore, we applied Min-Max normalization to all metrics to bring them to a comparable scale. This resulted in 30 performance metrics per model (10 datasets $\times 3$ metrics between 0 and 1). We then combined the scores (algorithm 1), as no model was strictly best in all 30 metrics. This allowed us to assign a scalar performance metric to each model as the sum of its 30 normalized performances to identify the best model on average.

Algorithm 1: Computation of Accumulated, Normalized Performance

Input: Performance Tensor $P \in \mathbb{R}^{\mathcal{M} \times D \times M}$
$\mathcal{M}$ is the number of metrics, D is the number of datasets, and M is the number of models.
for m ***in*** $\mathcal{M}$ **do**
  for d ***in*** D **do**
    $d_{min} \leftarrow \min P[m, d, :]$
    $d_{max} \leftarrow \max P[m, d, :]$
    $P[m, d, :] \leftarrow \frac{P[m,d,:] - d_{min}}{d_{max} - d_{min}}$
  end
end
$output \leftarrow [\sum P[:, :, m]$ **for** m **in** $M]$
return *output*

In early experiments, we investigated the influence of model depth on performance. We found that an 8-layered GIFFLAR model performed best, on average, on downstream tasks (see Figure 1A).

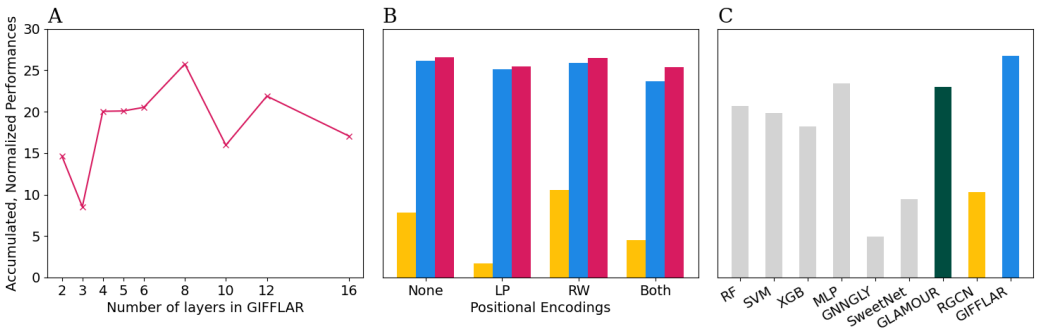

Figure 1: Averaged normalized performances (ANPs) for different model comparisons. The ANPs are calculated over all models within one subplot. Elements with the same color refer to the same model. A: ANPs for the GIFFLAR model with 1024 feature dimensions with different depths. B: ANPs comparing RGCN with 128 feature dimensions (leftmost in each group) to two GIFFLAR models with 128 (middle) and 1024 feature dimensions (rightmost), respectively. The bars are grouped by added positional encodings (PEs). C: ANPs comparing GIFFLAR to the eight baselines.

Frac.	Immun.	Glycos.	D	K	P	C	O	F	G	S
full	.8145	.9655	.9531	.9214	.8481	.7877	.6464	.6236	.5381	.4943
OOD	1	.6902	.9247	.9198	.7952	.7234	.5507	.5192	.4381	.4043

Table 2: Matthews Correlation Coefficient of the final GIFFLAR architecture on the test set, establishing a new SOTA. Performances on the full test set and on only out-of-distribution (OOD) glycans are shown. Performances of all models on the validation set can be found in Table A1.

We also investigated how node feature dimensions and positional encodings (PEs) impacted model performance. We tested 128-dimensional and 1024-dimensional random features combined with RandomWalk PEs and Laplacian PEs. Neither individual nor combined PEs consistently improved model performance (see Figure 1B). Further, we observed almost equal performance of 128- and 1024-dimensional feature vectors. Following the principle of Occam's Razor, we thus favored 128-dimensional features, which were then scaled up to 1024 dimensions within the GNN layers. In Figure A3, we report a comparison of the unnormalized scores.

After determining the optimal model architecture, we compared this model to retrained baselines based on Morgan 1024-bit Fingerprints (Random Forests, Support Vector Machines, GradientBoosting, and a Feedforward Neural Network), homogeneous graphs (GNNGLY on an atomic level, and SweetNet and GLAMOUR on a topological level), and RGCN on heterogeneous graphs. Here, again, our new GIFFLAR architecture showed superior performance, being the best-in-class model despite a lower number of trainable parameters (see Figure 1C and Tables A2 and A1). The fingerprint-based baselines appear very strong, which is a common phenomenon observed in the field [26].

Table 2 shows the performance of GIFFLAR on the held-out test set, i. e., data that did not influence the model development. The out-of-distribution (OOD) data performance may vary from this, as shown in [27]. Therefore, we report the performance of the OOD fraction of the test set separately. In Appendix B.1, we describe which glycans we considered OOD. Notably, the variance of the OOD-MCC is relatively high for the Immunogenicity and Glycosylation datasets, as the test set only contained 31 and 12 OOD samples, respectively. The test sets of the taxonomy datasets contained over 400 OOD samples and are, therefore, more stable.

5 Discussion, Conclusion, Limitations, and Future Work

Here, we presented GIFFLAR, a novel graph neural network architecture designed specifically for glycan representation learning. By leveraging combinatorial complexes and higher-order message passing, GIFFLAR achieves state-of-the-art performance across a wide range of glycan property prediction tasks. GIFFLAR consistently outperforms existing methods, including traditional ma-

chine learning approaches using Morgan fingerprints, homogeneous graph neural networks such as GNNGLY, SweetNet, and GLAMOUR, and heterogeneous graph models such as RGCN. The superior performance of GIFFLAR can likely be attributed to its ability to capture both atomic-level structure and topological information of glycans simultaneously. Due to its superior performance, we expect that GIFFLAR will become the new foundation in computational glycobiology and yield improvements in relevant downstream tasks. Another advantage of GIFFLAR over the baselines is that the feature extractor can be trained end-to-end specifically for a task, e. g., in lectin-glycan interaction prediction.

Using combinatorial complexes allows GIFFLAR to represent glycans at multiple levels of abstraction—atoms, bonds, and monosaccharides—in a single, unified framework. This multi-level representation, combined with higher-order message passing, enables the model to learn rich, hierarchical features crucial for accurate glycan property prediction. While we focused on glycans in this work, the principles behind GIFFLAR could be extended and adapted to other complex biomolecules, such as metabolites or lipids. Exploring these applications could further broaden the impact of our approach.

GIFFLAR achieves high predictive performance, yet interpreting its learned representations remains challenging. Developing techniques to visualize and explain model predictions would be valuable for glycobiologists and could lead to new insights into glycan structure-function relationships. We further anticipate that future work could explore the value of pre-training such models on larger sets of unlabeled data, similar to established procedures in protein representation learning.

In conclusion, GIFFLAR presents a significant advance in glycan representation learning, demonstrating the power of combining combinatorial complexes with higher-order message passing in graph neural networks. As glycomics continues to grow in importance within the life sciences, we envision that GIFFLAR and its future extensions will play a crucial role in unlocking the full potential of glycan-related research and applications. An example application of GIFFLAR as a trainable feature extractor for glycans is the field of lectin-glycan interaction prediction.

Acknowledgments and Disclosure of Funding

R.J. was supported by the Knut and Alice Wallenberg Foundation and the University of Gothenburg. This work was funded by a Branco Weiss Fellowship - Society in Science awarded to D.B., the Knut and Alice Wallenberg Foundation, and the University of Gothenburg, Sweden.

The authors thank the Drug Bioinformatics group at the Helmholtz Institute for Pharmaceutical Research Saarland, Saarbrücken, Germany, for providing the computing resources to conduct the experiments.

References

[1] Ajit Varki. Biological roles of glycans. *Glycobiology*, 27(1):3–49, 2016.

[2] Salomé S. Pinho, Ines Alves, Joana Gaifem, and Gabriel A. Rabinovich. Immune regulatory networks coordinated by glycans and glycan-binding proteins in autoimmunity and infection. *Cellular Molecular Immunology*, 20(10):1101–1113, 2023.

[3] Jon Lundstrøm and Daniel Bojar. Structural insights into host–microbe glycointeractions. *Current Opinion in Structural Biology*, 73:102337, 2022.

[4] Linh Nguyen, Kelli A McCord, Duong T Bui, Kim M Bouwman, Elena N Kitova, Mohamed Elaish, Dhanraj Kumawat, Gour C Daskhan, Ilhan Tomris, Ling Han, et al. Sialic acid-containing glycolipids mediate binding and viral entry of sars-cov-2. *Nature Chemical Biology*, 18(1):81–90, 2022.

[5] Mariana C Silva, Ângela Fernandes, Maria Oliveira, Carlos Resende, Alexandra Correia, Julio C de Freitas-Junior, Aonghus Lavelle, Jéssica Andrade-da Costa, Magdalena Leander, Helena Xavier-Ferreira, et al. Glycans as immune checkpoints: removal of branched n-glycans enhances immune recognition preventing cancer progression. *Cancer immunology research*, 8(11):1407–1425, 2020.

[6] Christopher J Day, Paola Favuzza, Sabrina Bielfeld, Thomas Haselhorst, Leonie Seefeldt, Julia Hauser, Lucy K Shewell, Christian Flueck, Jessica Poole, Freda E-C Jen, et al. The essential malaria protein pfcyrpa targets glycans to invade erythrocytes. *Cell Reports*, 43(4), 2024.

[7] Daniel Bojar and Frederique Lisacek. Glycoinformatics in the artificial intelligence era. *Chemical Reviews*, 122(20):15971–15988, 2022.

[8] Rebekka Burkholz, John Quackenbush, and Daniel Bojar. Using graph convolutional neural networks to learn a representation for glycans. *Cell Reports*, 35(11), 2021.

[9] Alhasan Alkuhlani, Walaa Gad, Mohamed Roushdy, and Abdel-Badeeh M Salem. Gnngly: Graph neural networks for glycan classification. *IEEE Access*, 11:51838–51847, 2023.

[10] Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, Tolga Birdal, Tamal K Dey, Soham Mukherjee, Shreyas N Samaga, et al. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:2206.00606*, 2022.

[11] Minghao Xu, Yunteng Geng, Yihang Zhang, Ling Yang, Jian Tang, and Wentao Zhang. Glycanml: A multi-task and multi-structure benchmark for glycan machine learning. *arXiv preprint arXiv:2405.16206*, 2024.

[12] Daniel Bojar, Rani K Powers, Diogo M Camacho, and James J Collins. Deep-learning resources for studying glycan-mediated host-microbe interactions. *Cell Host & Microbe*, 29(1):132–144, 2021.

[13] Davide Alocci, Julien Mariethoz, Alessandra Gastaldello, Elisabeth Gasteiger, Niclas G Karlsson, Daniel Kolarich, Nicolle H Packer, and Frédérique Lisacek. Glyconnect: glycoproteomics goes visual, interactive, and analytical. *Journal of proteome research*, 18(2):664–677, 2018.

[14] Luc Thomès, Rebekka Burkholz, and Daniel Bojar. Glycowork: A python package for glycan data science and machine learning. *Glycobiology*, 31(10):1240–1244, 2021.

[15] Roman Joeres, Daniel Bojar, and Olga V Kalinina. Glyles: Grammar-based parsing of glycans from iupac-condensed to smiles. *Journal of Cheminformatics*, 15(1):37, 2023.

[16] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15*, pages 593–607. Springer, 2018.

[17] Jon Lundstrøm, Emma Korhonen, Frédérique Lisacek, and Daniel Bojar. Lectinoracle: a generalizable deep learning model for lectin–glycan binding prediction. *Advanced Science*, 9(1):2103807, 2022.

[18] Benjamin P. Kellman, Julien Mariethoz, Yujie Zhang, Sigal Shaul, Mia Alteri, Daniel Sandoval, Mia Jeffris, Erick Armingol, Bokan Bao, Frederique Lisacek, Daniel Bojar, and Nathan E. Lewis. Decoding glycosylation potential from protein structure across human glycoproteins with a multi-view recurrent neural network. *bioRxiv*, 2024.

[19] Somesh Mohapatra, Joyce An, and Rafael Gómez-Bombarelli. Chemistry-informed macromolecule graph representation for similarity computation, unsupervised and supervised learning. *Machine Learning: Science and Technology*, 3(1):015028, 2022.

[20] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[21] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[22] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[23] Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. *PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA*, 1974.

[24] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[25] Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):1–21, 2012.

[26] Dejun Jiang, Zhenxing Wu, Chang-Yu Hsieh, Guangyong Chen, Ben Liao, Zhe Wang, Chao Shen, Dongsheng Cao, Jian Wu, and Tingjun Hou. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13:1–23, 2021.

[27] Roman Joeres, David B Blumenthal, and Olga V Kalinina. Datasail: Data splitting against information leakage. *bioRxiv*, pages 2023–11, 2023.

[28] Yam Eitan, Yoav Gelberg, Guy Bar-Shalom, Fabrizio Frasca, Michael Bronstein, and Haggai Maron. Topological blind spots: Understanding and extending topological deep learning through the lens of expressivity. *arXiv preprint arXiv:2408.05486*, 2024.

[29] Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. *Advances in neural information processing systems*, 34:2625–2640, 2021.

[30] Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning*, pages 1026–1037. PMLR, 2021.

[31] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*, 2021.

[32] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023.

[33] Luc Thomès, Rebekka Burkholz, and Daniel Bojar. Glycowork: A python package for glycan data science and machine learning. *Glycobiology*, 31(10):1240–1244, 2021.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[36] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[37] William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019.

[38] Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh Jha, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. Torchmetrics-measuring reproducibility in pytorch. *Journal of Open Source Software*, 7(70):4101, 2022.

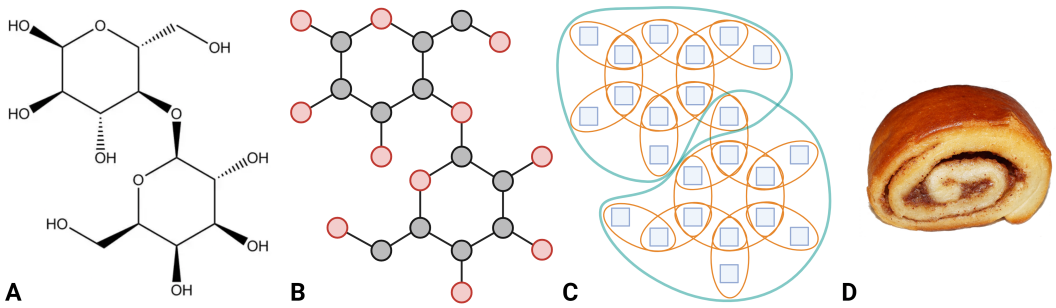

Figure A1: Lactose in three abstractions. Figure A shows the chemical formula, and Figure B shows the graph of atoms (GOA). Figure C shows the combinatorial complex and cells of all ranks. 0-cells in blue represent the atoms; 1-cells in orange are bonds between atoms, and 2-cells in green show whole monosaccharides. Figure D shows an inspirational and motivational Gifflar, a Swedish cinnamon roll.

A Theory

This section will explain how we interpret glycans as graphs and use combinatorial complexes and higher-order message passing to compute their topology-aware embeddings.

Combinatorial Complexes Combinatorial complexes (CCs) are the abstract superclass of simplicial complexes, cellular complexes, and more. As they are central to our work, we will precisely define CCs following the definition of Eitan et al. [28]. Definitions of simplicial and cellular complexes are given in [29, 30].

Definition A.1. A combinatorial complex (CC) is a tuple $(S, \mathcal{X}, f)$ where S is a set, $\mathcal{X} \subseteq \mathcal{P}(S) \setminus \emptyset$ is a subset of its powerset $\mathcal{P}(S)$, and $f \coloneqq \mathcal{X} \mapsto \mathbb{Z}_0^+$ is a function with the following properties:

$$\forall s \in S.\ \{s\} \in \mathcal{X} \tag{3}$$

$$\forall x, y \in \mathcal{X}.\ x \subset y \implies f(x) < f(y), \text{ i. e., } f \text{ is order-preserving.} \tag{4}$$

The elements of S correspond to a graph's nodes (or vertices); elements of $\mathcal{X}$ are called cells. Different cell ranks are based on the organization level within the combinatorial complex. In a general CC, all cells of rank i form the i-skeleton, denoted as $\mathcal{X}_i$. For our work, nodes are 0-cells (representing atoms), edges between nodes are 1-cells (representing bonds), and 2-cells are collections of edges (representing monomers).

Neighborhood functions Within and between ranks of a CC, $C = (S, \mathcal{X}, f)$, one can define a neighborhood function $\mathcal{N} \coloneqq \mathcal{X} \mapsto \mathcal{P}(\mathcal{X})$. For $x \in \mathcal{X}_i$, the four neighborhood functions relevant to our work are:

$$\mathcal{B}_{i,j}(x) = \{y \in \mathcal{X}_i \mid \exists z \in \mathcal{X}_j \text{ s.t. } x, y \subseteq z\} \tag{5}$$

$$\mathcal{C}_{i,j}(x) = \{y \in \mathcal{X}_i \mid \exists z \in \mathcal{X}_j \text{ s.t. } z \subseteq x, y\} \tag{6}$$

$$\mathcal{N}^{\downarrow}_{i,j}(x) = \{y \in \mathcal{X}_j \mid x \subseteq y\} \tag{7}$$

$$\mathcal{N}^{\uparrow}_{i,j}(x) = \{y \in \mathcal{X}_j \mid y \subseteq x\} \tag{8}$$

where i and j are different ranks of a CC. Equations 5 and 6 define neighborhoods within rank i over shared cells with higher (Eq. 5) and lower (Eq. 6) ranks, respectively. Conversely, Equations 7 and 8 define neighborhoods between ranks, where (Eq. 7) is the set of higher-rank neighbors and (Eq. 8) is the set of lower-rank neighbors. Figure A1 visualizes all definitions and relations on the example of lactose.

Higher-order message passing Using the neighborhoods on CCs defined above, we can define a message-passing scheme on CCs following Hajij et al. [10].

Definition A.2. Let $\mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_n\}$ be a set of neighborhood functions defined on a CC $\mathcal{X}$, e. g., $\mathcal{N} = \{\mathcal{B}_{0,1}, \mathcal{C}_{1,0}, \mathcal{N}^{\downarrow}_{0,1}, \mathcal{N}^{\uparrow}_{1,0}\}$. Furthermore, $\mathbf{h}^l_x$ is the embedding of cell $x \in \mathcal{X}$ in layer

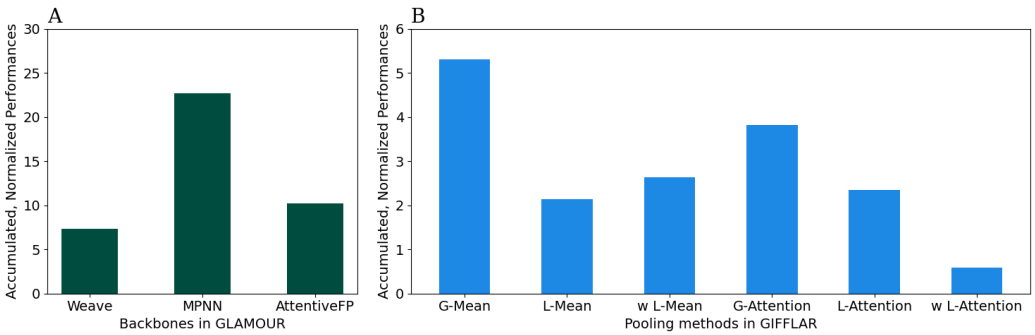

Figure A2: A: Comparison of different backbone architectures in GLAMOUR compared on all 10 datasets. B: Comparison of different pooling methods in GIFFLAR as described in B. This was conducted only on the Immunogenicity and Taxonomy-Domain datasets for runtime reasons.

$l \in \{0, \ldots, L\}$. Higher-order message passing on $\mathcal{X}$ is defined via this update rule:

$$\mathbf{h}_x^{l+1} = \sigma\left(\mathbf{h}_x^l, \bigotimes_{\mathcal{N}_k \in \mathcal{N}} \bigoplus_{y \in \mathcal{N}_k} \theta_{\mathcal{N}_k}^l \left(\mathbf{h}_x^l, \mathbf{h}_y^l\right)\right) \tag{9}$$

Here, σ is a non-linear activation function, $\theta_{\mathcal{N}_k}^l$ is a layer-specific differentiable function, e. g., neural networks, and $\bigotimes$ and $\bigoplus$ are permutation-invariant aggregation functions gathering inter-neighborhoods and intra-neighborhoods, respectively.

For the final readout into a single embedding representing $\mathcal{X}$, the readout function is defined as follows:

$$h_{out} = \sigma\left(\bigotimes_{i=0}^{l} \bigoplus_{x \in \mathcal{C}} \theta_i \left(\mathbf{h}_x^L\right)\right) \tag{10}$$

with l being the number of ranks in $\mathcal{X}$ and L the number of layers to compute. Everything else is as defined for Equation 9.

Positional Encodings In graphs, node features can be computed based on their connectivity and neighborhoods within the graph. Therefore, one can compute so-called *Positional Encodings*. In this work, we experimented with two types of PEs: RandomWalk PEs and Laplacian PEs. k-dimensional RandomWalk PEs are computed from a k-step random walk through the graph, starting from each node. After each step i, the number of walkers in a node is extracted as a i-th feature for the positional encoding [31]. k-dimensional Laplacian PEs are defined as the first k dimensions of the eigenvectors of the graph's Laplacian with randomly assigned signs [32].

B Further Ablation Studies

As Mohapatra et al. do not name one backbone architecture as the best among the five provided, we compared the usable ones. The provided implementation does not allow applying GCN— and GAT-based models to graphs representing monomeric molecules. In Figure A2A, we compared the Weave, MPNN, and AttentiveFP models, among which the MPNN performed best and was selected as the GLAMOUR backbone for the baseline model in this work.

Figure A2B compares different pooling mechanisms for the GIFFLAR model with 128-dimensional feature vectors. Here, we only compared the models on the Glycosylation, Immunogenicity, and Domain datasets to reduce runtime. The six modes we compared were:

- **Global Mean**: Computing the mean over all embeddings in layer L regardless of their cell rank.

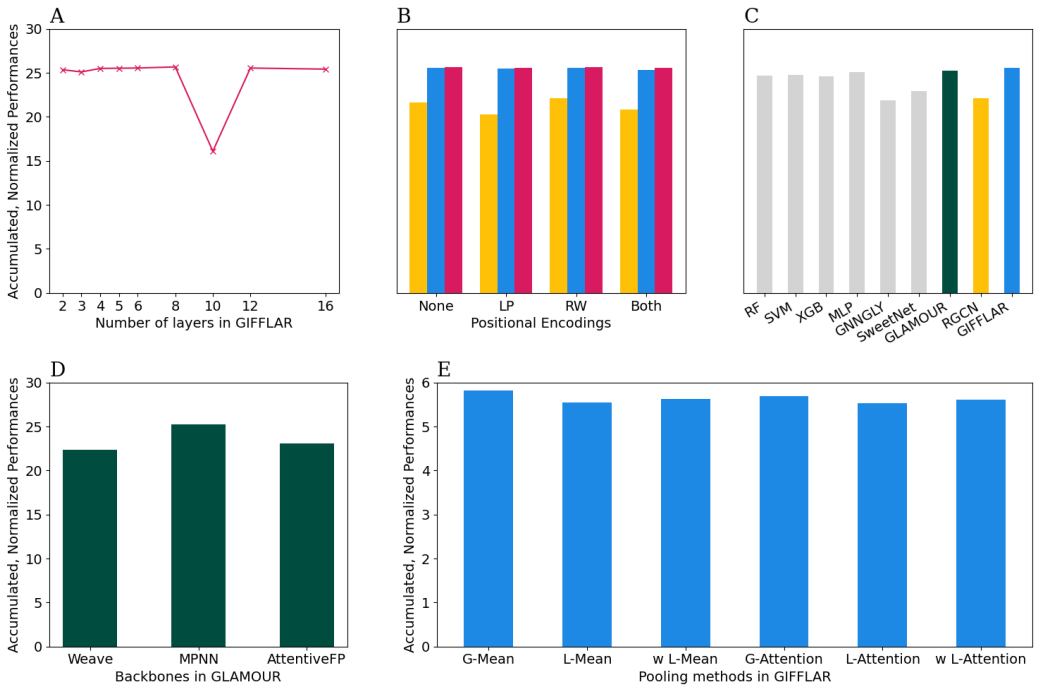

Figure A3: Reevaluation of Figures 1 and A2 without the normalization in Algorithm 1.

- **Local Mean**: First, compute the mean over all embeddings in Layer L within each cell rank and then take the mean over the three rank means.
- **weighted Local Mean**: Same as Local Mean, but multiply each cell rank with a learnable weight before summation.
- **Global Attention**: Computing the Soft Attention over all embeddings in layer L regardless of their cell rank.
- **Local Attention**: First, compute the Soft Attention over all embeddings in Layer L within each cell rank and then take the mean over the three rank means.
- **weighted Local Attention**: Same as Local Attention, but multiply each cell rank with a learnable weight before summation.

Surprisingly, attention-free pooling functions generally performed better than attention-based ones in this analysis. Further, global poolings were better than local or weighted local methods. We conclude that global mean pooling seems to constitute the most performant pooling operation for GIFFLAR.

B.1 Out-Of-Distribution Glycans

To compute the similarity between glycans, we used the `annotate` module from glycowork [33] to compute fingerprints for each glycan based on their monosaccharide and disaccharide motifs. We then defined OOD glycans as those glycans in the test set that had a fingerprint with a Tanimoto similarity of below 0.75, compared to the closest glycan in the training or validation split.

C Model Comparison

To investigate the impact of Algorithm 1 on the performance comparison in Figures 1 and A2, we report the same plot but unnormalized in Figure A3 (except for MCC, which was transformed by $x \mapsto \frac{x+1}{2}$ to be in the interval `[0,1]`). Compared to Figures 1 and A2, the ordering of the models by performance does not change significantly, but the differences between the models shrink.

Model	Immun.	Glycos.	D	K	P	C	O	F	G	S
Morgan Fingerprint (1024-bit)										
RF	.8223	.9648	.9129	.8749	.8010	.7094	.5546	.4944	.4613	.4439
SVM	.8034	.9648	.8793	.8403	.7466	.6398	.4588	.4369	.4137	.3880
XGB	.8302	.9824	.8718	.8348	.7393	.6282	.4705	.4420	.3870	.3467
MLP	.8481	.8704	.9106	.8763	.8033	.7206	.5413	.5097	.4708	.4282
Homogeneous Graphs										
GNNGLY	.5328	.7611	.7717	.7747	.6609	.4685	.0156	.0150	.0151	.0154
SweetNet	.7590	.8784	.8841	.7704	.6232	.5288	.0156	.1872	.0151	.1175
GLAMOUR	**.9212**	.9767	.9111	.8704	.7864	.6857	.4998	.4785	.4320	.4407
Heterogeneous Graphs										
RGCN	.6954	.0000	.8810	.8409	.7211	.4039	.2288	.0314	.2530	.0194
GIFFLAR	.8930	**.9883**	**.9298**	**.9011**	**.8278**	**.7714**	**.6118**	**.5795**	**.5391**	**.4898**

Table A1: Comparison of Matthews Correlation Coefficient of GIFFLAR to the seven baselines. Bold values mark the best performances on a certain dataset. The single-letter column names refer to the taxonomic levels of D–domain, K–kingdom, P–phylum, C–class, O–order, F–family, G–genus, and S–species. We will also use these in all other tables.

Model	Training time	∅ # nodes per graph	# param.
Morgan Fingerprint (1024-bit)			
RF	0.1 min	–	–
SVM	2.2 min	–	–
XGB	1.7 min	–	–
MLP	11.7 min	–	527K
Homogeneous Graphs			
GNNGLY	20.5 min	91	8M
SweetNet	22.7 min	13	37.2M
GLAMOUR	9.5 min	7	18.2M
Heterogeneous Graphs			
RGCN_{RW}	55.3 min	195	41.9M
GIFFLAR	45.0 min	195	35.1M

Table A2: Measurements of models on Domain Dataset (as an exemplary dataset).

We report Matthews Correlation Coefficients (MCC) for the final models and baselines to give more perspective on the model performances (Table A1). For the multi-label datasets, the MCC was computed per label and then averaged over the labels unweighted.

D Training Details

Table A2 lists some core measurements of the models we compare. We used the Domain prediction from the taxonomy collection as an example dataset. The table's average number of nodes per graph is computed on the training set. The first three models (RF, SVM, and XGB) were trained using scikit-learn v1.5.1 [34] on a regular CPU. The deep learning models were trained using PyTorch v2.3.1 [35], PyTorch Geometric v2.5.3 [36], and PyTorch Lightning v2.3.2 [37] on an NVIDIA GeForce RTX 3090 with 24GB GPU RAM. All metrics for all models were computed using TorchMetrics v1.4.0 [38].

The code and the data splits are available on `github.com/BojarLab/GIFFLAR`.