Representation Invariance of GNNs: Going Beyond Isomorphism

Abstract

Graph Neural Networks (GNNs) leverage the topology of graphs to learn informative representations. Since isomorphic graphs represent the same information, there has been a significant effort to develop GNNs that return the same results over isomorphic graphs. It is known in the data management community that the same data can be represented in distinct nonisomorphic forms, e.g., normalization and denormalization of relational or graph data. In this paper, we postulate that GNNs should be invariant against these types of variations in data representations. We formalize the notion of invariance using concepts in schema management, e.g., mapping and constraints. We report our preliminary results, which indicate that GNNs are not often robust under these variations.

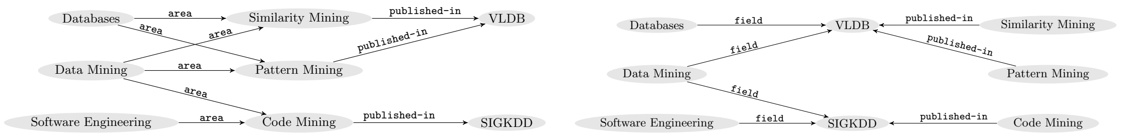
1. Introduction

Graph Neural Networks (GNNs) are widely used to solve various predictive problems in graphs Masci et al. (2016); Hamilton (2020); Yang et al. (2020). They map the features and relationships of the nodes in a graph to a rich vector representation of the graph, which can be used for downstream ML tasks. Since isomorphic graphs represent the same information, GNNs aim to learn the same representation for isomorphic graphs Masci et al. (2016); Hamilton (2020); Huang et al. (2023), i.e., they are invariant to permutation of nodes in the graph.

However, research on data management has shown that the same information can be represented in different nonisomorphic structures Hull (1984); Arenas (2006); Chodpathumwan et al. (2021). Classic examples of such variations are (de)normalizations in relational and XML data where the original and normalized databases represent the same information under different structures Hull (1984); Arenas (2006). Consider Figures 1(a) and 1(b) that show fragments of DBLP (*dblp.org*) and SIGMOD Record (*sigmod.org/publications*) bibliographic graphs, respectively. These graphs are not isomorphic but contain the same information on the same set of *papers*, *conferences*, and *research areas*. DBLP connects each paper directly to its research areas and conferences. Given that all papers in a conference share the same set of research areas, the graph in Figure 1(b) represents this relationship by connecting the research areas of a paper to its conferences. As another example, developers may add additional links to a graph to connect nodes that are often queried together to increase the querying efficiency.

The results of ML tasks must be the same for the input datasets (training and testing) that contain the same information. Thus, a GNN should learn the same representation over the data graphs that contain the same information. Since current GNNs use the topology in a graph to learn its representation, they can learn different representations over nonisomorphic graphs that contain the same information, e.g., graphs in Figures 1(a) and 1(b). For example, they may learn different representations for *VLDB* conference in the graphs Figures 1(*a*) and 1(*b*). This may cause the downstream node classification task to place the *VLDB* conference in different classes over each dataset. Thus, GNN-based methods may deliver accurate results in some styles of graph representation and perform poorly in others. As GNN applications grow rapidly, they will face more variations in graph representations, which increases the impacts of this shortcoming on the effectiveness of GNN-based methods.

In this paper, we formalize the problem of representation invariance of GNNs and the space of reasonable representational variations. We report our preliminary empirical results on the representation invariance of some current GNN architectures.



(a) Fragments of DBLP

(b) Fragments of SIGMOD Record

Figure 1: Two graph datasets on about papers, conferences and research areas.

2. Representation Invariance

2.1. Data Model

Let $\mathcal{L}$ and $\mathbf{V}$ be countably finite sets of labels and node ids, respectively. A *graph* G over $\mathcal{L}$ is a directed graph (V, E) in which V is a subset of $\mathbf{V}$ and $E \subseteq V \times L \times V$. We denote an edge from node u to node v whose label is a as (u, a, v). We associate each node with a real-valued vector of dimensions d, i.e., its features, in $\mathcal{R}^d$. If all nodes and edges of the graph share the same label, the graph is often called homogeneous and heterogeneous otherwise Hamilton (2020).

A schema $\mathcal{G}$ is a pair $(\mathcal{L}, \Sigma)$ in which $\mathcal{L}$ is a (finite) set of labels and Σ is a finite set of *constraints*. Constraints restrict the instances of a schema and are often expressed as logical formulas Boneva et al. (2015); Arenas (2006). Constraints are provided by the user or discovered from the data. For example, a popular group of constraints is *tuple-generating dependencies* (*tgd*). A tgd on schema $(\mathcal{L}, \Sigma)$ is in the form of $\forall \bar{x}(\phi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))$ where $\bar{x}$ and $\bar{y}$ are sets of variables, and ϕ and ψ are logical formulas in a query language over $\mathcal{L}$. For example, the graph in Figure 1(*a*) satisfies the constraint $(x_1, \texttt{area}, x_3) \wedge (x_3, \texttt{pub-in}, x_4)$ $\wedge (x_2, \texttt{pub-in}, x_4) \rightarrow (x_1, \texttt{area}, x_2)$. Each *instance* G of schema $\mathcal{G} = (\mathcal{L}, \Sigma)$ is a graph with node and edge labels of $\mathcal{L}$ such that all the constraints in Σ are held in the graph, i.e., $G \models \Sigma$. We denote the set of all instances of $\mathcal{G}$ by $\mathrm{Inst}(\mathcal{G})$.

2.2. Mappings

Researchers have used the idea of *invertible mappings* to formalize the notion of information equivalence for different representations of data Hull (1984); Fagin (2007); Chodpathumwan et al. (2021).Intuitively, if we can map the information stored in one structure to another structure and recover the information, these structures represent equivalent information. We formally state this idea as follows. A *mapping* from schema $\mathcal{G}$ to schema $\mathcal{H}$ is a function of $\mathrm{Inst}(\mathcal{G})$ to $\mathrm{Inst}(\mathcal{H})$, which maps each graph of $\mathcal{G}$ to a graph of $\mathcal{H}$. For example, a mapping from the schema of the graph whose fragments are shown in Figure 1(a) to the schema of the one in Figure 1(b) eliminates the edges between `research areas` and `papers` and connects `research areas` to the `venues` of those `papers`. We use the closed-world semantic for our mapping in which τ maps instances of $\mathcal{G}$ to instances of $\mathcal{H}$ whose information is produced only using τ.

The mapping τ from $\mathcal{G}$ to $\mathcal{H}$ is *invertible* if there is a mapping τ^{-1} such that, for each instance $G \in \mathrm{Inst}(\mathcal{G})$, τ^{-1} maps every instance $\tau(G)$ to G and only G. We call τ^{-1} the *inverse* of τ. If there is an invertible mapping from $\mathcal{G}$ to $\mathcal{H}$, we can map every instance represented under $\mathcal{G}$ to an instance under $\mathcal{H}$ and recover it without losing any information. For example, consider mapping $\tau_{1a,1b}$ from the schema of the graph in Figure 1(a) to the schema of the graph in Figure 1(b). The constraint in Figure 1(a) implies that the papers published in the same conference are related to the same set of research areas. Hence, one may change the structure shown in Figure 1(a) such that the research areas associated with a paper are instead connected to the paper via the conference of the paper and get the graph in Figure 1(b). The information in the original graph can be recovered using the information in Figure 1(b). Mappings may be expressed as a set of rules in a graph query language Barcelo et al. (2013); Boneva et al. (2015).

2.3. Invariance & Equivariance

GNNs learn low-dimensional representations of graphs that should exhibit properties expressed in graph topology and features. The learned representation is often expressed as d-dimension feature vectors on the nodes in the graph. Given schema $\mathcal{G}$, we denote this by the function $f : \mathrm{Inst}(\mathcal{G}) \rightarrow V \times \mathcal{R}^d$. For each node v in the input graph of f, R contains a tuple with the node id of v and vice versa. The function f can learn representations with a larger or smaller dimensionality than d. For simplicity of exposition, we assume that the learned vector has the same dimensionality as the feature vectors on the nodes. Our definitions extend to other cases.

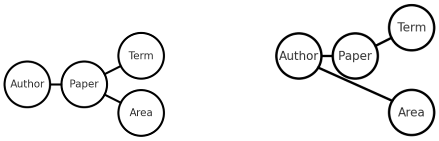
The function f is *equivariant* under invertible mapping τ if for every $G \in \mathrm{Inst}(\mathcal{G})$ there is a one-to-one mapping π between $f(G)$ and $f(\tau(G))$ such that for all $r \in f(G)$, we have $\pi(r) = r$. That is, f learns the same representation for every corresponding node in G and $\tau(G)$. For certain tasks, such as graph classifications, GNNs are modified to return a single value, e.g., by aggregating representations of relevant nodes. We can model these cases by the function $f : \mathrm{Inst}(\mathcal{G}) \rightarrow \mathcal{R}$. We call the function f *invariant* under invertible mapping τ if for every $G \in \mathrm{Inst}(\mathcal{G})$, we have $f(G) = f(\tau(G))$.

Current GNNs use the topology of the graph to learn representations Hamilton (2020). Because invertible mappings can considerably modify the topologies of graphs, current GNN architectures are not invariant or equivariant. For example, GNNs that use message passing approach share and aggregate features between neighbors of each node to learn its representation. Since invertible mappings like the one in Figure 1 can change the neighbors of a node, the GNNs based on the message passing technique may not be invariant or equivariant under these mappings.

3. Preliminary Empirical Results

Dataset. We use the DBLP variant of Fu et al. (2020), a heterogeneous graph with 4,057 authors, 14,328 papers, 7,723 terms and 20 conferences. The learning task is node classification of authors' research area.

Dataset Variation. We alter the original graph Fu et al. (2020) by replacing conference nodes with area nodes. This variation allows us to construct invertible graph transformations 2. The learning task is to predict conference for each paper.



(*a*) DBLP1 (*b*) DBLP2

Figure 2: Schematic variations of DBLP data.

Methods. We conduct experiments on two GNN architectures: RGCN Schlichtkrull et al. (2018) and MAGNN Fu et al. (2020). We follow the metapath definitions from Fu et al. (2020), using paper as both the start and end node for each metapath.

Metrics. We evaluate using precision/recall/F1 score. To assess *representational invariance*, we introduce a robustness metric that measures prediction consistency, i.e., whether the model assigns the same class to each node across both DBLP variants.

Table 1: DBLP evaluation results on the test set.

Model	**Dataset**	**Precision**	**Recall**	**F1**	**Robustness**
RGCN	DBLP1	0.5123	0.5120	0.5111	0.7749
RGCN	DBLP2	0.4970	0.4970	0.4948	
MAGNN	DBLP1	0.5630	0.5202	0.6109	0.6809
MAGNN	DBLP2	0.5265	0.5253	0.5272	

Results. MAGNN attains higher precision/recall/F1 than RGCN on both DBLP variants (Table 1), likely due to its metapath-based use of higher-order structure. Neither model shows strong representational invariance. RGCN is relatively more stable than MAGNN, suggesting explicit metapath encoding increases sensitivity to structural variations. Training is faster for RGCN (5 min) than MAGNN (10 min) due to the latter's attention over metapaths.

4. Ongoing Work

We plan to develop GNN architectures that are invariant to information-preserving transformations by considering and formally defining sets of popular transformations across graph datasets.

References

Marcelo Arenas. Normalization theory for xml. *SIGMOD Rec.*, 35(4):57–64, December 2006. ISSN 0163-5808. doi: 10.1145/1228268.1228284. URL http://doi.acm.org/10.1145/1228268.1228284.

Pablo Barcelo, Jorge Perez, and Juan Reutter. Schema mappings and data exchange for graph databases. In *ICDT*, 2013.

I. Boneva, A. Bonifati, and R. Ciucanu. Graph data exchange with target constraints. In *EDBT/ICDT Workshop GraphQ*, PODS '17, pages 171–176, 2015. ISBN 978-1-4503-4198-1.

Yodsawalai Chodpathumwan, Arash Termehchy, Stephen A. Ramsey, Aayam Shrestha, Amy Glen, and Zheng Liu. Structural generalizability: The case of similarity search. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, page 326–338, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383431. doi: 10.1145/3448016.3457316. URL https://doi.org/10.1145/3448016.3457316.

Ronald Fagin. Inverting schema mappings. *ACM Trans. Database Syst.*, 32(2), 2007.

Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the web conference 2020*, pages 2331–2341, 2020.

William L Hamilton. *Graph representation learning*. Morgan & Claypool Publishers, 2020.

Ningyuan (Teresa) Huang, Ron Levie, and Soledad Villar. Approximately equivariant graph networks. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

R. Hull. Relative information capacity of simple relational database schemata. 1984.

Jonathan Masci, Emanuele Rodolà, Davide Boscaini, Michael M Bronstein, and Hao Li. Geometric deep learning. In *SIGGRAPH ASIA 2016 Courses*, pages 1–50. 2016.

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.

Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. Heterogeneous network representation learning: A unified framework with survey and benchmark. *IEEE Transactions on Knowledge and Data Engineering*, 34(10):4854–4873, 2020.