

Interpretable Graph Networks Formulate Universal Algebra Conjectures

Francesco Giannini*
Università di Siena, Italy
francesco.giannini@unisi.it

Stefano Fioravanti*
Università di Siena, Italy
stefano.fioravanti@unisi.it

Oguzhan Keskin
University of Cambridge, UK
ok313@cam.ac.uk

Alisia Maria Lupidi
University of Cambridge, UK
am1201@cam.ac.uk

Lucie Charlotte Magister
University of Cambridge, UK
lcm67@cam.ac.uk

Pietro Lió
University of Cambridge, UK
pl219@cam.ac.uk

Pietro Barbiero*
University of Cambridge, UK
pb737@cam.ac.uk

Abstract

The rise of Artificial Intelligence (AI) recently empowered researchers to investigate hard mathematical problems which eluded traditional approaches for decades. Yet, the use of AI in Universal Algebra (UA)—one of the fields laying the foundations of modern mathematics—is still completely unexplored. This work proposes the first use of AI to investigate UA’s conjectures with an equivalent equational and topological characterization. While topological representations would enable the analysis of such properties using graph neural networks, the limited transparency and brittle explainability of these models hinder their straightforward use to empirically validate existing conjectures or to formulate new ones. To bridge these gaps, we propose a general algorithm generating AI-ready datasets based on UA’s conjectures, and introduce a novel neural layer to build fully interpretable graph networks. The results of our experiments demonstrate that interpretable graph networks: (i) enhance interpretability without sacrificing task accuracy, (ii) strongly generalize when predicting universal algebra’s properties, (iii) generate simple explanations that empirically validate existing conjectures, and (iv) identify subgraphs suggesting the formulation of novel conjectures.

1 Introduction

Universal Algebra (UA, (6)) is one of the foundational fields of modern mathematics, yet the complexity of studying abstract algebraic structures hinders scientific progress and discourages many academics. Recently, the emergence of powerful AI technologies empowered researchers to investigate intricate mathematical problems which eluded traditional approaches for decades, leading to the solution of open problems (e.g., (22)) and discovery of new conjectures (e.g., (7)). Yet, universal algebra currently remains an uninvestigated realm for AI, a completely uncharted territory with deep impact in all mathematical disciplines.

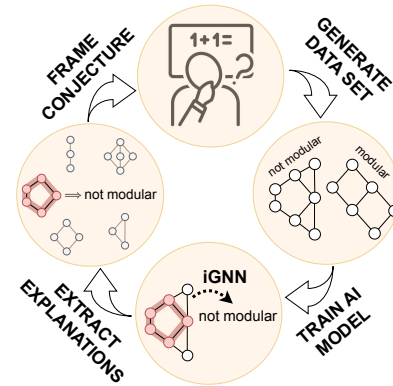


Figure 1: Interpretable graph networks support universal algebra research.

Universal algebra studies algebraic structures from an abstract perspective. Interestingly, several UA conjectures equivalently characterize algebraic properties using equations or graphs (15). In theory, studying UA properties as graphs would enable the use of powerful AI techniques, such as Graph Neural Networks (GNN, (34)), which excel on graph-structured data. However, two factors currently limit scientific progress. First, the *absence of benchmark datasets* suitable for machine learning prevents widespread application of AI to UA. Second, *GNNs’ opaque reasoning* obstructs human understanding of their decision process (33). Compounding the issue of GNNs’ limited transparency, GNN explainability methods mostly rely on brittle and untrustworthy local/post-hoc methods (13; 25; 26; 33; 39) or pre-defined subgraphs for explanations (2; 36), which are often unknown in UA.

Contributions. In this work, we investigate universal algebra’s conjectures through AI (Figure 1), venturing for the first time in this previously uncharted territory. Our work includes three significant contributions. First, we propose a novel algorithm that generates a dataset suitable for training AI models based on an UA equational conjecture. Second, we generate and release the first-ever universal algebra’s dataset compatible with AI, which contains more than 29,000 lattices and the labels of 5 key properties i.e., modularity, distributivity, semi-distributivity, join semi-distributivity, and meet semi-distributivity. And third, we introduce a novel neural layer that makes GNNs fully interpretable, according to Rudin’s (33) notion of interpretability. The results of our experiments demonstrate that interpretable GNNs (iGNNs): (i) enhance GNN interpretability without sacrificing task accuracy, (ii) strongly generalize when trained to predict universal algebra’s properties, (iii) generate simple concept-based explanations that empirically validate existing conjectures, and (iv) identify subgraphs which could be relevant for the formulation of novel conjectures. Our findings demonstrate the potential of our methodology and open the doors of universal algebra to AI.

2 Background

Universal Algebra is a branch of mathematics studying general and abstract algebraic structures. *Algebraic structures* are typically represented as ordered pairs $\mathbf{A} = (A, F)$, consisting of a non-empty set A and a collection of operations F defined on the set. UA aims to identify algebraic properties (often in equational form) shared by various mathematical systems. In particular, *varieties* are classes of algebraic structures sharing a common set of identities, which enable the study of algebraic systems based on their common properties. Prominent instances of varieties that have been extensively studied across various academic fields encompass Groups, Rings, Boolean Algebras, Fields, and many others. A particularly relevant variety of algebras are Lattices (details in Appendix A.4), which are often studied for their connection with logical structures.

Definition 2.1. A *lattice* \mathbf{L} is an algebraic structure composed by a non-empty set L and two binary operations \vee and \wedge , satisfying the commutativity, associativity, idempotency, and absorption axioms.

Equivalently a lattice can be characterized as a partially ordered set in which every pair of elements has a *supremum* and an *infimum* (cf. Appendix A.4). Lattices also have formal representations as graphs via *Hasse diagrams* (L, E) (e.g., Figure 2), where each node $x \in L$ is a lattice element, and directed¹ edges $(x, y) \in E \subseteq L \times L$ represent the ordering relation, such that if $(x, y) \in E$ then $x \leq_L y$ in the ordering of the lattice. A *sublattice* \mathbf{L}' of a lattice \mathbf{L} is a lattice such that $L' \subseteq L$ and \mathbf{L}' preserves the original order (the “essential structure”) of L , i.e. for all $x, y \in L'$ then $x \leq_{L'} y$ if and only if $x \leq_L y$. The foundational work by Birkhoff (5), Dedekind (9), and Jónsson (17) played a significant role in discovering that some significant varieties of lattices can be characterized through the omission of one or more lattices. Specifically, a variety \mathcal{V} of lattices is said to *omit* a lattice \mathbf{L} if it cannot be identified as a sublattice of any lattice in \mathcal{V} . A parallel line of work in UA characterizes lattices in terms of equational (“*term*₁ \approx *term*₂”) and quasi-equational (“if *equation*₁ holds then *equation*₂ holds”) properties, such as distributivity and modularity.

Definition 2.2. Let \mathbf{L} be a lattice. \mathbf{L} is *modular* if it satisfies $x \leq y \rightarrow x \vee (y \wedge z) \approx y \wedge (x \vee z)$; *distributive* if it satisfies $x \vee (y \wedge z) \approx (x \vee y) \wedge (x \vee z)$.

For instance, as showed in Figure 2, \mathbf{N}_5 is neither modular nor distributive- considering the substitution $x = a, y = c, z = b$. The same substitution shows that \mathbf{M}_3 is not distributive. The classes

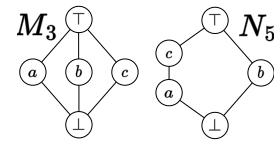


Figure 2: Hasse diagrams.

¹The orientation of Hasse diagrams is always to be meant bottom-up, hence we will omit arrows for simplicity.

of distributive and modular lattices show classical examples of varieties that can equivalently be characterized using equations and lattice omissions, as illustrated by the following theorems.

Theorem 2.3 (Dedekind (9)). *A lattice variety \mathcal{V} is modular if and only if \mathcal{V} omits \mathbf{N}_5 .*

Theorem 2.4 (Birkhoff (5)). *A lattice variety \mathcal{V} is distributive if and only if \mathcal{V} omits \mathbf{N}_5 and \mathbf{M}_3 .*

Starting from these classic results, the investigation of lattice omissions and the structural characterizations of classes of lattices has evolved into a rich and extensively studied field (15), but it was never approached with advanced AI methods before.

3 Methods

The problem of characterizing lattice varieties through lattice omission is very challenging as it requires the analysis of large (potentially infinite) lattices (5; 9; 17). To address this task, we propose the first AI-assisted framework supporting mathematicians in finding empirical evidences to validate existing conjectures and to suggest novel theorems. To this end, we propose a general algorithm (Section 3.1) allowing researchers in universal algebra to define a property of interest and generate a dataset suitable to train AI models. We then introduce interpretable graph networks (Section 3.2) which can suggest candidate lattices whose omission is responsible for the satisfaction of the given algebraic property.

3.1 A Tool to Generate Datasets of Lattice Varieties

We propose a general methodology to investigate any algebraic property whose validity can be verified on a finite lattice. In this work, we focus on properties that can be characterized via equations and quasi-equations. To train AI models, we propose a general dataset generator² for lattice varieties (Algorithm 1). The generator takes as input the number of nodes n in the lattices and a function to check whether a lattice satisfies a given property. We generate $2^{n \times n}$ matrices of size $n \times n$, containing all binary functions definable on $\{1, \dots, n\}^2$, and filter only binary matrices representing partial orders³. Then, we verify that the partial ordered set L is a lattice, by checking that any pair of nodes always has a unique infimum and supremum. This directly verifies that \wedge_L and \vee_L satisfy Definition 2.1. Finally, we check whether the lattice satisfies the target property or not, and append it and the property label to our dataset. We remark that checking the validity of a single ternary equation on a medium-size lattice is not computationally prohibitive (i.e., it “only” requires checking n^3 identities), but the number of existing lattices increases exponentially as n increases. For instance, it is known that there are at least 2,000,000 non-isomorphic lattices with $L = 10$ elements (4). Therefore, we only sample a fixed number of lattices per cardinality starting from a certain node cardinality. While this may seem a strong bias, we notice that known and relevant lattice omissions often rely on lattices with few nodes (5; 9). To empirically verify that this is not a significant limitation, in our experiments we deliberately investigate the generalization capacity of GNNs when trained on small-size lattices and tested on larger ones. This way we can use GNNs to predict the satisfiability of equational properties on large graph structures without explicitly checking them. Using Algorithm 1, we generated the first large-scale AI-compatible datasets of lattices containing more than 29,000 graphs and the labels of 5 key properties of lattice (quasi-)varieties i.e., modularity, distributivity, semi-distributivity, join semi-distributivity, and meet semi-distributivity, whose definitions can be found in Appendix A.

Algorithm 1: Generate dataset of lattice varieties.

```

Input:  $n \geq 1, hasProperty(\cdot, \cdot)$  //  $n$ : cardinality
Dataset = []
AllFuncs  $\leftarrow$  genAllFuncs( $n$ ) // binary functions as  $n \times n$  matrices
for  $L \in$  AllFuncs do //  $L(i, j) = 1$  meaning  $i \leq_L j$ 
  if isPartialOrder( $L$ ) then // check if  $\leq_L$  is refl., antisym. and trans.
    if isLattice( $L$ ) then // check if  $L$  is a lattice
      for  $i, j \leq n$  do
         $\wedge_L[i, j] \leftarrow \sup_{x \leq_L} \{x \leq_L i \text{ and } x \leq_L j\}$ 
         $\vee_L[i, j] \leftarrow \inf_{x \leq_L} \{i \leq_L x \text{ and } j \leq_L x\}$ 
      if hasProperty( $L, \wedge_L, \vee_L$ ) then // check  $\wedge_L, \vee_L$  properties
        Dataset.append( $[L, \text{True}]$ )
      else
        Dataset.append( $[L, \text{False}]$ )

```

3.2 Interpretable Graph Networks (iGNNs)

In this section, we design an interpretable graph network (iGNN, Figure 3) that satisfies the notion of "interpretability" introduced by Rudin (33). According to this definition, a machine learning

²The dataset generator code and the generated datasets will be made public in case of paper acceptance.

³Our algorithm optimizes this step considering only reflexive and antisymmetric binary relations, and enforces transitivity with an easy fix-point calculation.

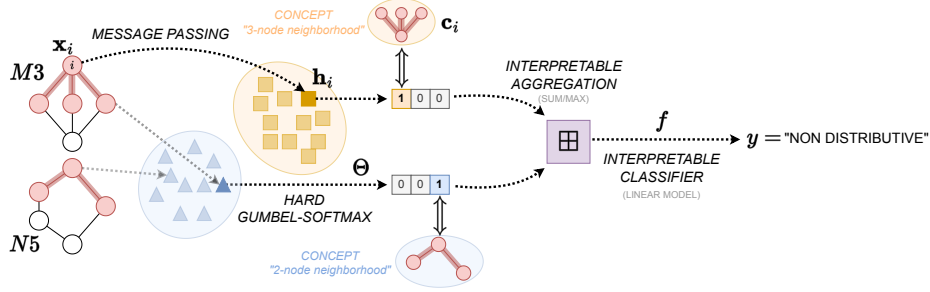


Figure 3: An interpretable graph layer (i) aggregates node features with message passing, (ii) generates a node-level concept space with a hard Gumbel-Softmax activation Θ , (iii) generates a graph-level concept space with an interpretable permutation invariant pooling function \boxplus on node-level concepts, and (iv) predicts a task label with an interpretable classifier f using graph-level concepts.

(ML) system is interpretable if and only if (1) its inputs are semantically meaningful, and (2) its model inference is simple for humans to understand (e.g., sparse and/or symbolic). This definition covers ML systems that take tabular datasets or sets of concepts as inputs, and (piece-wise) linear models such as logistic regression or decision trees. To achieve this goal in GNNs, we introduce an interpretable graph layer that learns semantically meaningful concepts and uses them as inputs for a simple linear classification layer. We then show how this layer can be included into existing architectures or into hierarchical iGNNs, which consist of a sequence of interpretable graph layers.

3.2.1 Interpretable Graph Layer

The interpretable graph layer (Figure 3) serves three main functions: message passing, concept generation, and task predictions. The first step of the interpretable graph layer involves a standard message passing operation (Eq. 1 right), which aggregates information from node neighbors. This operation enables to share and process relational information across nodes and it represents the basis of any GNN layer.

Node-level concepts. An interpretable concept space is the first step towards interpretability. Following Ghorbani et al. (10), a relevant concept is a “high-level human-understandable unit of information” shared by input samples and thus identifiable with clustering techniques. Message passing algorithms do cluster node embeddings based on the structure of node neighborhoods, as observed by Magister et al. (26). However, the real-valued large embedding representations $\mathbf{h}_i \in \mathbb{R}^q$, $q \in \mathbb{N}$ generated by message passing can be challenging for humans to interpret. To address this, we use a hard Gumbel-Softmax activation $\Theta : \mathbb{R}^q \mapsto \{0, 1\}^q$, following Azzolin et al. (2):





$$\mathbf{c}_i = \Theta(\mathbf{h}_i) \quad \mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j \in N_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right) \quad (1)$$

where ψ and ϕ are learnable functions aggregating information from a node neighborhood N_i , and \bigoplus is a permutation invariant aggregation function (such as sum or mean). During the forward pass, the Gumbel-Softmax activation Θ produces a one-hot encoded representation of each node embedding. Since nodes sharing the same neighborhood have similar embeddings \mathbf{h}_i due to message passing, they will also have the same one-hot vector \mathbf{c}_i due to the Gumbel-Softmax, and vice versa - we can then interpret nodes having the same one-hot concept \mathbf{c}_i as nodes having similar embeddings \mathbf{h}_i and thus sharing a similar neighborhood. More formally, we can assign a semantic meaning to a reference concept $\gamma \in \{0, 1\}^q$ by visualizing concept prototypes corresponding to the inverse images of a node concept vector. In practice, we can consider a subset of the input lattices Γ corresponding to the node’s (p -hop) neighborhood covered by message passing:






$$\Gamma(\gamma, p) = \left\{ \mathbf{L}^{(i,p)} \mid i \in L \wedge \mathbf{L} \in \mathcal{D} \wedge \mathbf{c}_i = \gamma \right\} \quad (2)$$

where \mathcal{D} is the set of all training lattices, and $\mathbf{L}^{(i,p)}$ is the graph corresponding to the p -hop neighborhood ($p \in \{1, \dots, |L|\}$) of the node $i \in L$, as suggested by Ghorbani et al. (10); Magister et al. (26). This way, by visualizing concept prototypes as subgraph neighborhoods, the meaning of the concept

representation becomes easily interpretable to humans (Figure 3), aiding in the understanding of the reasoning process of the network.

Example 3.1 (Interpreting node-level concepts). Consider the problem of classifying distributive lattices with a simplified dataset including \mathbf{N}_5 and \mathbf{M}_3 only, and where each node has a constant feature $x_i = 1$. As these two lattices only have nodes with 2 or 3 neighbours, one layer of message passing will then generate only two types of node embeddings e.g., $\mathbf{h}_{II} = [0.2, -0.4, 0.3]$ for nodes with a 2-nodes neighborhood (e.g., ) , and $\mathbf{h}_{III} = [0.6, 0.2, -0.1]$ for nodes with a 3-nodes neighborhood (e.g., ) . As a consequence, the Gumbel-Softmax will only generate two possible concept vectors e.g., $\mathbf{c}_{II} = [0, 0, 1]$ and $\mathbf{c}_{III} = [1, 0, 0]$. Hence, for instance the concept  belongs to \mathbf{c}_{II} , while  belongs to \mathbf{c}_{III} .

Graph-level concept embeddings. To generate a graph-level concept space in the interpretable graph layer, we can utilize the node-level concept space produced by the Gumbel-Softmax. Normally, graph-level embeddings are generated by applying a permutation invariant aggregation function on node embeddings. However, in iGNNs we restrict the options to (piece-wise) linear permutation invariant functions in order to follow our interpretability requirements dictated by Rudin (33). This restriction still includes common options such as max or sum pooling. Max pooling can easily be interpreted by taking the component-wise max over the one-hot encoded concept vectors \mathbf{c}_i . After max pooling, the graph-level concept vector has a value of 1 at the k -th index if and only if at least one node activates the k -th concept i.e., $\exists i \in L, \mathbf{c}_{ik} = 1$. Similarly, we can interpret the output of a sum pooling: a graph-level concept vector takes a value $v \in \mathbb{N}$ at the k -th index after sum pooling if and only if there are exactly v nodes activating the k -th concept i.e., $\exists i_0, \dots, i_v \in L, \mathbf{c}_{ik} = 1$.

Example 3.2 (Interpreting graph-level concepts). Following Example 3.1, let us use sum pooling to generate graph-level concepts. For an \mathbf{N}_5 graph, we have 5 nodes with exactly the same 2-node neighborhood. Therefore, sum pooling generates a graph-level embedding $[0, 0, 5]$, which certifies that we have 5 nodes of the same type e.g., . For an \mathbf{M}_3 graph, the top and bottom nodes have a 3-node neighborhood e.g., , while the middle nodes have a 2-node neighborhood e.g., . This means that sum pooling generates a graph-level embedding $[2, 0, 3]$, certifying that we have 2 nodes of type  and 3 nodes of type .

Interpretable classifier. To prioritize the identification of relevant concepts, we use a classifier to predict the task labels using the concept representations. A black-box classifier like a multi-layer perceptron (31) would not be ideal as it could compromise the interpretability of our model, so instead we use an interpretable linear classifier such as a single-layer network (19). This allows for a completely interpretable and differentiable model from the input to the classification head, as the input representations of the classifier are interpretable concepts and the classifier is a simple linear model which is intrinsically interpretable as discussed by Rudin (33). In fact, the weights of the perceptron can be used to identify which concepts are most relevant for the classification task. Hence, the resulting model can be used not only for classification, but also to interpret and understand the problem at hand.

3.2.2 Interpretable architectures

The interpretable graph layer can be used to instantiate different types of iGNNs. One approach is to plug this layer as the last message passing layer of a standard GNN architecture:

$$\hat{y} = f\left(\boxplus_{i \in K} \left(\Theta\left(\phi^{(K)}\left(\mathbf{h}_i^{(K-1)}, \bigoplus_{j \in N_i} \psi^{(K)}(\mathbf{h}_i^{(K-1)}, \mathbf{h}_j^{(K-1)})\right)\right)\right)\right) \quad (3)$$

$$\mathbf{h}_i^{(l)} = \phi^{(l)}\left(\mathbf{h}_i^{(l-1)}, \bigoplus_{j \in N_i} \psi^{(l)}(\mathbf{h}_i^{(l-1)}, \mathbf{h}_j^{(l-1)})\right) \quad l = 1, \dots, K \quad (4)$$

where f is an interpretable classifier (e.g., single-layer network), \boxplus is an interpretable piece-wise linear and permutation-invariant function (such as max or sum), Θ is a Gumbel-Softmax hard activation function, and $\mathbf{h}_i^0 = \mathbf{x}_i$. In this way, we can interpret the first part of the network as a feature extractor generating well-clustered latent representations from which concepts can be extracted. This

approach is useful when we only care about the most complex neighborhoods/concepts. Another approach is to generate a hierarchical transparent architecture where each GNN layer is interpretable:

$$\hat{y}^{(l)} = f\left(\boxplus_{i \in K} \left(\Theta\left(\mathbf{h}_j^{(l)}\right)\right)\right) \quad l = 1, \dots, K \quad (5)$$

In this case, we can interpret every single layer of our model with concepts of increasing complexity. The concepts extracted from the first layer represent subgraphs corresponding to the 1-hop neighborhood of a node, those extracted at the second layer will correspond to 2-hop neighborhoods, and so on. These hierarchical iGNNs can be useful to get insights into concepts with different granularities. By analyzing the concepts extracted at each layer, we gain a better understanding of the GNN inference and of the importance of different (sub)graph structures for the classification task.

3.2.3 Training

The choice of the activation and loss functions iGNNs depends on the nature of the task at hand and does not affect their interpretability. For classification tasks, we use standard activation functions such as softmax or sigmoid, along with standard loss functions like cross-entropy. For hierarchical iGNNs (HiGNNs), we apply the loss function at each layer of the concept hierarchy, as their layered architecture enables intermediate supervisions. This ensures that each layer is doing its best to extract the most relevant concepts to solve the task. Internal losses can also be weighted differently to prioritize the formation of optimal concepts of a specific size, allowing the HiGNN to learn in a progressive and efficient way.

4 Experimental Analysis

4.1 Research questions

In this section we analyze the following research questions:

- **Generalization** - Can GNNs generalize when trained to predict universal algebra’s properties? Can interpretable GNNs generalize as well?
- **Interpretability** - Do interpretable GNNs concepts empirically validate universal algebra’s conjectures? How can concept-based explanations suggest novel conjectures?

4.2 Setup

Baselines. For our comparative study, we evaluate the performance of iGNNs and their hierarchical version against equivalent GNN models (i.e., having the same hyperparameters such as number layers, training epochs, and learning rate). For vanilla GNNs we resort to common practice replacing the Gumbel-Softmax with a standard leaky ReLU activation. We exclude from our main baselines prototype or concept-based GNNs pre-defining graph structures for explanations, as for most datasets these structures are unknown. Appendix B covers implementation details. We show more extensive results including local and post-hoc explanations in Appendix ??.

Evaluation. We employ three quantitative metrics to assess a model’s generalization and interpretability. We use the Area Under the Receiver Operating Characteristic (AUC ROC) curve to assess task generalization. We evaluate generalization under two different conditions: with independently and identically distributed train/test splits, and out-of-distribution by training on graphs up to eight nodes, while testing on graphs with more than eight nodes (“strong generalization”). We further assess generalization under binary and multilabel settings (classifying 5 properties of a lattice at the same time). To evaluate interpretability, we use standard metrics such as completeness (38) and fidelity (32). Completeness⁴ assesses the quality of the concept space on a global scale using an interpretable model to map concepts to tasks, while fidelity measures the difference in predictions obtained with an interpretable surrogate model and the original model. Finally, we evaluate the meaningfulness of our concept-based explanations by visualizing and comparing the generated concepts with ground truth lattices like e.g. \mathbf{M}_3 and \mathbf{N}_5 , whose omission is known to be significant for modular and distributive properties. All metrics in our evaluation, across all experiments, are computed on test sets using 5 random seeds, and reported using the mean and 95% confidence interval.

⁴We assess the recall of the completeness as the datasets are very unbalanced towards the negative label.

5 Key Findings

5.1 Generalization

iGNNs improve interpretability without sacrificing task accuracy (Figure 4). Our experimental evaluation reveals that interpretable GNNs are able to strike a balance between completeness and fidelity, two crucial metrics that are used to assess generalization-interpretability trade-offs (32). We observe that the multilabel classification scenario, which requires models to learn a more varied and diverse set of concepts, is the most challenging and results in the lowest completeness scores on average. We also notice that the more challenging out-of-distribution scenario results in the lowest completeness and fidelity scores across all datasets. More importantly, our findings indicate that iGNNs achieve optimal fidelity scores, as their classification layer consists of a simple linear function of the learnt concepts which is intrinsically interpretable (33). On the contrary, interpretable surrogate models of black-box GNNs exhibit, as expected, lower fidelity scores, confirming analogous observations in the explainable AI literature (32; 33). In practice, this discrepancy between the original black-box predictions and the predictions obtained with an interpretable surrogate model questions the actual usefulness of black-boxes when interpretable alternatives achieve similar results in solving the problem at hand, as extensively discussed by Rudin (33). Overall, these results demonstrate how concept spaces are highly informative to solve universal algebra’s tasks and how the interpretable graph layer may improve GNNs’ interpretability without sacrificing task accuracy. We refer the reader to Appendix D for detailed discussion on quantitative analysis of concept space obtained by iGNNs under different generalization settings with comparisons to their black-box counterparts.

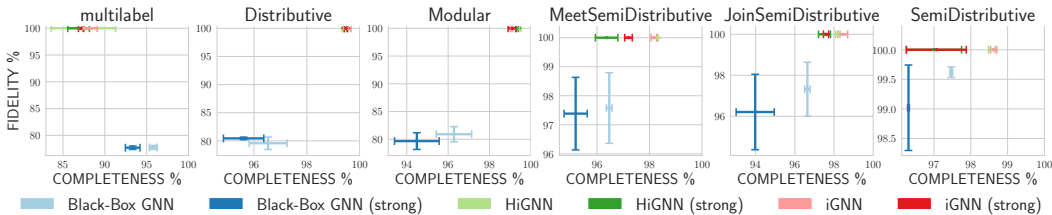


Figure 4: Accuracy-interpretability trade-off in terms of concept completeness (accuracy) and model fidelity (interpretability). iGNNs attain optimal fidelity as model inference is inherently interpretable, outmatching equivalent black-box GNNs. All models attain similar results in terms of completeness.

GNNs strongly generalize on universal algebra’s tasks (Figure 5). Our experimental findings demonstrate the strong generalization capabilities of GNNs across the universal algebra tasks we designed. Indeed, we stress GNNs test generalization abilities by training the models on graphs of size up to n (with n ranging from 5 to 8), and evaluating their performance on much larger graphs of size up to 50. We designed this challenging experiment in order to understand the limits and robustness of interpretable GNNs when facing a significant data distribution shift from training to test data. Remarkably, iGNNs exhibit robust generalization abilities (similar to their black-box counterparts) when trained on graphs up to size 8 and tested on larger graphs. This evidence confirms the hypothesis that interpretable models can deliver reliable and interpretable predictions, as suggested by Rudin (33). However, we observe that black-box GNNs slightly outperform iGNNs when trained on even smaller lattices.

We hypothesize that this is due to the more constrained architecture of iGNNs, which imposes tighter bounds on their expressiveness when compared to standard black-box GNNs. Notably, training with graphs of size up to 5 or 6 significantly

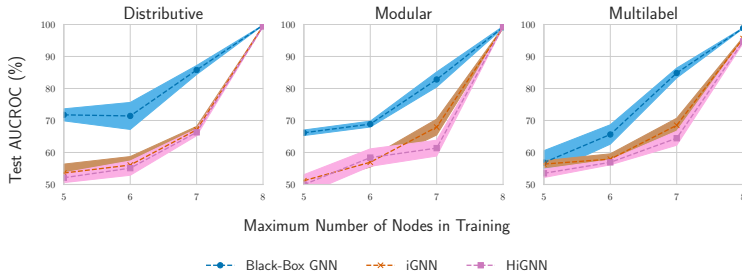


Figure 5: Strong generalization performance with respect to the maximum number of nodes used in training.

diminishes GNNs generalization in the tasks we designed. We hypothesize that this is due to the scarcity of non-distributive and non-modular lattices during training, but it may also suggest that some patterns of size 7 and 8 might be quite relevant to generalize to larger graphs. Unfortunately, running generalization experiments with $n \leq 4$ was not possible since all such lattices trivially omitted N_5 and M_3 . It is worth mentioning that GNNs performed well even in the challenging multilabel case, where they had to learn a wider and more diverse set of concepts and tasks. In all experiments, we observe a plateau of the AUC ROC scores for $n = 8$, thus suggesting that a training set including graphs of this size might be sufficient to learn the relevant patterns allowing the generalization to larger lattice varieties. For detailed numerical results across all tasks, we refer the reader to Table 1 in Appendix C. Overall, these results emphasize the potential of GNNs in addressing complex problems in universal algebra, providing an effective tool to handle lattices that are difficult to analyze manually with pen and paper.

5.2 Interpretability

Concept-based explanations empirically validate universal algebra’s conjectures (Figure 6).

We present empirical evidence to support the validity of theorems 2.3 and 2.4 by examining the concepts generated for modular and distributive tasks. For this investigation we leverage the interpretable structure of iGNNs. Similarly to Ribeiro et al. (32), we visualize in Figure 6 the weights of our trained linear classifier representing the relevance of each concept. We remark that the visualization is limited to the (top-5) most negative weights, as we are interested in those concepts that negatively affect the prediction of a property. In the same plot, we also show the prototype of each concept represented by the 2-hop neighborhood of a node activating the concept, following a similar procedure as Ghorbani et al. (10); Magister et al. (26); Azzolin et al. (2). Using this visualization, we investigate the presence of certain concepts when classifying modular and distributive lattices.

For the modularity task, our results show that the lattice N_5 appears among non-modular concepts, but is never found in modular lattices, while the lattice M_3 appears among both modular and non-modular concepts, which is consistent with Theorem 2.3. In the case of distributivity, we observe that both M_3 and N_5 are present among non-distributive concepts, and are never found in distributive lattices, which is also in line with Theorem 2.4. These findings provide a large-scale empirical evidence for the validity of theorems 2.3 and 2.4, and further demonstrate the effectiveness of graph neural networks in learning and analyzing lattice properties. Overall, these results highlight how interpretable GNNs can not only learn the properties of universal algebra but also identify structures that are unique to one type of lattice (e.g., non-modular) and absent from another (e.g., modular), thus providing human-interpretable explanations for what the models learn.

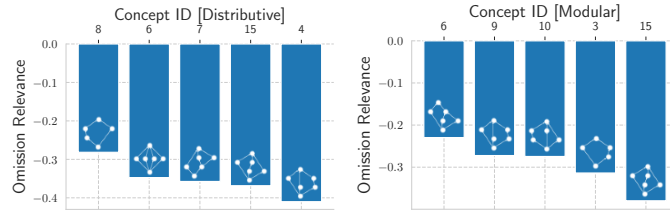


Figure 6: Ranking of relevant clusters of lattices (x-axis) according to the interpretable GNN linear classifier weights (y-axis, the lower the more relevant the cluster). N_5 is always the most important lattice to omit for modularity, while both M_3 and N_5 are relevant for distributivity, thus validating theorems 2.3 and 2.4.

Contrastive explanations highlight topological differences between properties of lattice varieties (Figure 7).

We leverage interpretable GNNs to analyze the key topological differences of classical lattice properties such as join and meet semi-distributivity characterized by relevant quasi-equations (cf. Appendix A.6). To this end, we visualize specific concept prototypes corresponding to lattices that are not meet semi-distributive against lattices that are meet semi-distributive. We observe N_5 but not M_3 among the concepts of meet semi-distributive lattices, while we observe both N_5 and M_3 only in concepts that are not meet semi-distributive. This observation suggests that N_5 is not a key lattice for meet semi-distributive

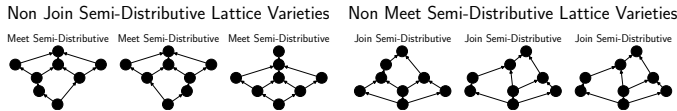


Figure 7: Contrastive explanations showing lattice varieties with a pair of discording labels to highlight the key difference between join and meet semi-distributivity.

lattices, *unlike distributive lattices*. Furthermore, we find that the lattice pattern \wp is relevant for non meet semidistributivity, while its dual \wp^* is relevant for non join semidistributivity, thus empirically confirming the hypotheses of Jónsson and Rival (18). These findings are significant because they demonstrate how analyzing concepts in interpretable GNNs can provide universal algebraists with a powerful and automatic tool to formulate new conjectures based on identifying simple lattices that play a role in specific properties. By leveraging the power of interpretable GNNs, we may uncover previously unknown connections between different properties and identify new patterns and structures that could lead to the development of new conjectures and theorems in universal algebra, providing exciting opportunities for future research in universal algebra.

6 Discussion

Relations with Graph Neural Network explainability. Graph Neural Networks (GNNs,(34)) process relational data generating node representations by combining the information of a node with that of its neighbors, thanks to a general learning paradigm known as message passing (11). A number of post-hoc explainability techniques have been proposed to explain the reasoning of GNNs. Inspired by vision approaches (35; 32; 10), early explainability techniques focused on feature importance (30), while subsequent works aimed to extract local explanations (39; 25; 36) or global explanations using conceptual subgraphs by clustering the activation space (26; 40; 27). However, all these techniques either rely on pre-defined subgraphs for explanations (which are often unknown in UA) or provide post-hoc explanations which may be brittle and unfaithful as extensively demonstrated by Rudin (33). On the contrary, our experiments show that iGNNs generate interpretable predictions according to Rudin (33) notion of interpretability via linear classifiers applied on sparse human-understandable concept representations.

Limitations. The approach proposed in this paper focuses on universal algebra conjectures characterized both algebraically and topologically. Our methodology is limited to finite lattices, which may not capture all relevant information about infinite algebraic structures. However, the insights gained from finite-lattice explanations can still provide valuable information regarding a given problem (albeit with potentially limited generalization). Moreover, our approach is restricted to topological properties on graphs, while non-structural properties may require the adoption of other kinds of (interpretable) models.

Broader impact and perspectives. AI techniques are becoming increasingly popular for solving previously intractable mathematical problems and proposing new conjectures (21; 24; 7; 12). However, the use of modern AI methods in universal algebra was a novel and unexplored field until the development of the approach presented in this paper. To this end, our method uses interpretable graph networks to suggest graph structures that characterize relevant algebraic properties of lattices. With our approach, we empirically validated Dedekind (9) and Birkhoff (5) theorems on distributive and modular lattices, by recovering relevant lattices. This approach can be readily extended—beyond equational properties determined by the omission of a sublattice in a variety (37)—to any structural property of lattices, including the characterization of congruence lattices of algebraic varieties (1; 20; 28; 37). Our methodology can also be applied (beyond universal algebra) to investigate (almost) any mathematical property that can be topologically characterized on a graph, such as the classes of graphs/diagraphs with a fixed set of polymorphisms (23; 3; 29). However, as universal algebra is a foundational branch of modern mathematics, any contribution to this field can already have significant implications in various mathematical disciplines.

Conclusion. This paper presents the first-ever AI-assisted approach to investigate equational and topological conjectures in the field of universal algebra. To this end, we present a novel algorithm to generate datasets suitable for AI models to study equational properties of lattice varieties. While topological representations would enable the use of graph neural networks, the limited transparency and brittle explainability of these models hinder their use in validating existing conjectures or proposing new ones. For this reason, we introduce a novel neural layer to build fully interpretable graph networks to analyze the generated datasets. The results of our experiments demonstrate that interpretable graph networks: enhance interpretability without sacrificing task accuracy, strongly generalize when predicting universal algebra’s properties, generate simple explanations that empirically validate existing conjectures, and identify subgraphs suggesting the formulation of novel conjectures. These

promising results demonstrate the potential of our methodology, opening the doors of universal algebra to AI with far-reaching impact across all mathematical disciplines.

References

- [1] Paolo Aglianò, Stefano Bartali, and Stefano Fioravanti. On Freese’s technique. *arXiv:2302.11452*, 2022.
- [2] Steve Azzolin, Antonio Longa, Pietro Barbiero, Pietro Liò, and Andrea Passerini. Global explainability of gnns via logic combination of learned concepts. *arXiv preprint arXiv:2210.07147*, 2022.
- [3] Libor Barto, Marcin Kozik, and Todd Niven. The csp dichotomy holds for digraphs with no sources and no sinks (a positive answer to a conjecture of bang-jensen and hell). *SIAM Journal on Computing*, 38(5):1782–1802, 2009.
- [4] Joel Berman and Paweł Idziak. Counting finite algebras in the post varieties. *International Journal of Algebra and Computation*, 10(03):323–337, 2000.
- [5] Garrett Birkhoff. On the structure of abstract algebras. In *Mathematical proceedings of the Cambridge philosophical society*, volume 31, pages 433–454. Cambridge University Press, 1935.
- [6] Stanley Burris and H. P. Sankappanavar. *A Course in Universal Algebra*. Springer, 1981.
- [7] Alex Davies, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli. Advancing mathematics by guiding human intuition with AI. *Nature*, 600(7887):70–74, December 2021. doi: 10.1038/s41586-021-04086-x. URL <https://doi.org/10.1038/s41586-021-04086-x>.
- [8] Alan Day. A characterization of modularity for congruence lattices of algebras*. *Canadian Mathematical Bulletin*, 12(2):167–173, 1969. doi: 10.4153/CMB-1969-016-6.
- [9] Richard Dedekind. Über die von drei Moduln erzeugte Dualgruppe. *Math. Ann.*, 1900.
- [10] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019.
- [11] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.
- [12] Yang-Hui He. Machine-learning mathematical structures. *International Journal of Data Science in the Mathematical Sciences*, pages 1–25, 2022.
- [13] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 2022.
- [14] Martin Hyland and John Power. The category theoretic understanding of universal algebra: Lawvere theories and monads. *Electronic Notes in Theoretical Computer Science*, 172:437–458, 2007. ISSN 1571-0661. doi: <https://doi.org/10.1016/j.entcs.2007.02.019>. URL <https://www.sciencedirect.com/science/article/pii/S1571066107000874>. Computation, Meaning, and Logic: Articles dedicated to Gordon Plotkin.
- [15] Peter Jipsen and Henry Rose. *Varieties of Lattices*. Springer Berlin, 1992.
- [16] Bjarni Jonnson. Algebras whose congruence lattices are distributive. *MATHEMATICA SCANDINAVICA*, 21:110–121, Dec. 1967. doi: 10.7146/math.scand.a-10850. URL <https://www.mscaand.dk/article/view/10850>.
- [17] Bjarni Jónsson. On the representation of lattices. *Mathematica Scandinavica*, 1953.

- [18] Bjarni Jónsson and Ivan Rival. Lattice varieties covering the smallest non-modular lattice variety. *Pacific J. Math.*, Volume 82, 1978.
- [19] Samandarov Erkaboy Karimboyevich and Abdurakhmonov Olim Nematullayevich. Single layer artificial neural network: Perceptron. *European Multidisciplinary Journal of Modern Science*, 5:230–238, Apr. 2022. URL <https://emjms.academicjournal.io/index.php/emjms/article/view/253>.
- [20] Keith Kearnes and Emil Kiss. The shape of congruence lattices. *Memoirs of the American Math. Soc.*, 2013.
- [21] Guillaume Lample and François Charton. Deep learning for symbolic mathematics. In *International Conference on Learning Representations*.
- [22] Guillaume Lample and François Charton. Deep learning for symbolic mathematics. *arXiv preprint arXiv:1912.01412*, 2019.
- [23] Barto Libor and Kozik Marcin. Absorbing Subalgebras, Cyclic Terms, and the Constraint Satisfaction Problem. *Logical Methods in Computer Science*, Volume 8, Issue 1, 2012.
- [24] Donald W Loveland. *Automated theorem proving: A logical basis*. Elsevier, 2016.
- [25] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.
- [26] Lucie Charlotte Magister, Dmitry Kazhdan, Vikash Singh, and Pietro Liò. Gcexplainer: Human-in-the-loop concept-based explanations for graph neural networks. *arXiv preprint arXiv:2107.11889*, 2021.
- [27] Lucie Charlotte Magister, Pietro Barbiero, Dmitry Kazhdan, Federico Siciliano, Gabriele Ciravegna, Fabrizio Silvestri, Mateja Jamnik, and Pietro Lio. Encoding concepts in graph neural networks. *arXiv preprint arXiv:2207.13586*, 2022.
- [28] J. B. Nation. Varieties whose congruences satisfy certain lattice identities. *Algebra Universalis*, 1974.
- [29] Miroslav Olšák. The local loop lemma. *Algebra universalis*, 2020.
- [30] Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10772–10781, 2019.
- [31] Marius-Constantin Popescu, Valentina Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8, 07 2009.
- [32] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.
- [33] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.
- [34] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.
- [35] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.
- [36] Minh Vu and My T Thai. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. *Advances in neural information processing systems*, 33:12225–12235, 2020.
- [37] Philip M Whitman. Free lattices. *Annals of Mathematics*, pages 325–330, 1941.

- [38] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565, 2020.
- [39] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks, 2019.
- [40] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. Protgnn: Towards self-explaining graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9127–9135, 2022.

A Algebra definitions

A.1 Formal definitions for Universal Algebra

Universal algebra is the field of mathematics that studies algebraic structures, which are defined as a set A along with its own collection of operations. An n -ary operation on A is a function that takes n elements of A and returns a single element from the set. More formally (6; 8; 16):

Definition A.1. N-ary function For A non-empty set and n nonnegative integer we define $A^0 = \{\emptyset\}$ and, for $n > 0$, A^n is the set of n -tuples of elements from A . An n -ary operation (or function) on A is any function f from A^n to A ; n is the arity (or rank) of f . An operation f on A is called an n -ary operation if its arity is n .

Definition A.2. Algebraic Structure An algebra \mathcal{A} is a pair $\langle A, F \rangle$ where A is a non-empty set called universe and F is a set of finitary operations on A .

Apart from the operations on A , an algebra is further defined by axioms, that in the particular case of universal algebras are often of the form of identities. The collection of algebraic structures defined by equational laws are called varieties. (14)

Definition A.3. Variety A nonempty class \mathbf{K} of algebras of type \mathcal{F} is called a variety if it is closed under subalgebras, homomorphic images, and direct products.

Definition A.4. A *lattice* \mathbf{L} is an algebraic structure composed by a non-empty set L and two binary operations \vee and \wedge satisfying the following axioms and their duals obtained exchanging \vee and \wedge :

$$\begin{array}{ll} x \vee y \approx y \vee x & \text{(commutativity)} \\ x \vee (y \vee z) \approx (x \vee y) \vee z & \text{(associativity)} \\ x \vee x \approx x & \text{(idempotency)} \\ x \approx x \vee (x \wedge y) & \text{(absorption)} \end{array}$$

Theorem A.5 ((6)). A partially ordered set L is a lattice if and only if for every $a, b \in L$ both supremum and infimum of $\{a, b\}$ exist (in L) with $a \vee b$ being the supremum and $a \wedge b$ the infimum.

Definition A.6. Let \mathbf{L} be a lattice. Then \mathbf{L} is *modular*(*distributive*, \vee -*semi-distributive*, \wedge -*semi-distributive*) if it satisfies the following equation:

$$\begin{array}{ll} x \leq y \rightarrow x \vee (y \wedge z) \approx y \wedge (x \vee z) & \text{(modularity)} \\ x \vee (y \wedge z) \approx (x \vee y) \wedge (x \vee z) & \text{(distributivity)} \\ x \vee y \approx x \vee z \rightarrow x \vee (y \wedge z) \approx x \vee y & \text{(\vee-semi-distributivity)} \\ x \wedge y \approx x \wedge z \rightarrow x \wedge (y \vee z) \approx x \wedge y & \text{(\wedge-semi-distributivity).} \end{array}$$

Furthermore a lattice \mathbf{L} is semi-distributive if is both \vee -semi-distributive and \wedge -semi-distributive

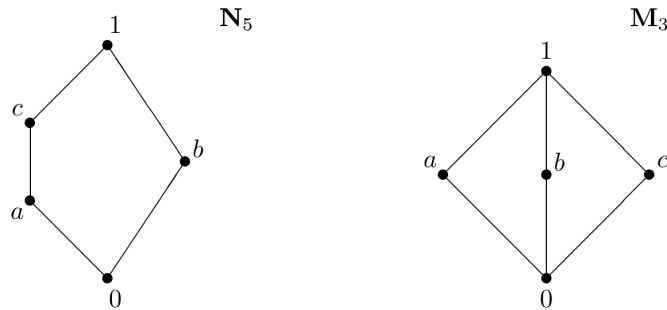


Figure 8: \mathbf{N}_5 , a non-modular non-distributive and \mathbf{M}_3 , a modular non-distributive lattice.

Congruence lattices of algebraic structures are partially ordered sets such that every pair of elements has unique supremum and infimum determined by the underlying algebra. This object is important relatively to algebraic structures’ properties, many of which can be described by omission or admission of certain subpatterns in a graph.

Definition A.7. Congruence Lattice

For every algebra \mathcal{A} on the set A , the identity relation on A , and $A \times A$ are trivial congruences. An algebra with no other congruences is called simple. Let $\text{Con}(\mathcal{A})$ be the set of congruences on the algebra \mathcal{A} . Because congruences are closed under intersection, we can define a meet operation: $\wedge : \text{Con}(\mathcal{A}) \times \text{Con}(\mathcal{A}) \rightarrow \text{Con}(\mathcal{A})$ by simply taking the intersection of the congruences $E_1 \wedge E_2 = E_1 \cap E_2$. Congruences are not closed under union, however we can define the closure operator of any binary relation E , with respect to a fixed algebra \mathcal{A} , such that it is a congruence, in the following way: $\langle E \rangle_{\mathcal{A}} = \bigcap \{F \in \text{Con}(\mathcal{A}) \mid E \subseteq F\}$. Note that the closure of a binary relation is a congruence and thus depends on the operations in \mathcal{A} , not just on the carrier set. Now define $\vee : \text{Con}(\mathcal{A}) \times \text{Con}(\mathcal{A}) \rightarrow \text{Con}(\mathcal{A})$ as $E_1 \vee E_2 = \langle E_1 \cup E_2 \rangle_{\mathcal{A}}$. For every algebra \mathcal{A} , $(\text{Con}(\mathcal{A}), \wedge, \vee)$ with the two operations defined above forms a lattice, called the congruence lattice of \mathcal{A} .

Definition A.8. Subalgebra Let \mathbf{A} and \mathbf{B} be two algebras of the same type. Then \mathbf{B} is a *subalgebra* of \mathbf{A} if $B \subseteq A$ and every fundamental operation of \mathbf{B} is the restriction of the corresponding operation of \mathbf{A} , i.e., for each function symbol f , $f^{\mathbf{B}}$ is $f^{\mathbf{A}}$ restricted to \mathbf{B} .

Definition A.9. Homomorphic image Suppose \mathbf{A} and \mathbf{B} are two algebras of the same type \mathcal{F} . A mapping $\alpha : A \rightarrow B$ is called a *homomorphism* from \mathbf{A} to \mathbf{B} if

$$\alpha f^{\mathbf{A}}(a_1, \dots, a_n) = f^{\mathbf{B}}(\alpha a_1, \dots, \alpha a_n)$$

for each n -ary f in \mathcal{F} and each sequence a_1, \dots, a_n from \mathbf{A} . If, in addition, the mapping α is onto then \mathbf{B} is said to be a homomorphic image of \mathbf{A} .

Definition A.10. Direct product Let \mathbf{A}_1 and \mathbf{A}_2 be two algebras of the same type \mathcal{F} . We define the direct product $\mathbf{A}_1 \times \mathbf{A}_2$ to be the algebra whose universe is the set $A_1 \times A_2$, and such that for $f \in \mathcal{F}$ and $a_i \in A_1, a'_i \in A_2, 1 \leq i \leq n$,

$$f^{\mathbf{A}_1 \times \mathbf{A}_2}(\langle a_1, a'_1 \rangle, \dots, \langle a_n, a'_n \rangle) = \langle f^{\mathbf{A}_1}(a_1, \dots, a_n), f^{\mathbf{A}_2}(a'_1, \dots, a'_n) \rangle$$

B Baselines’ details

In practice, we train all models using eight message passing layers and different embedding sizes ranging from 16 to 64. We train all models for 200 epochs with a learning rate of 0.001. For interpretable models, we set the Gumbel-Softmax temperature to the default value of 1 and the activation behavior to "hard," which generates one-hot encoded embeddings in the forward pass, but computes the gradients using the soft scores. For the hierarchical model, we set the internal loss weight to 0.1 (to score it roughly 10% less w.r.t. the main loss). Overall, our selection of baselines aims at embracing a wide set of training setups and architectures to assess the effectiveness and versatility of GNNs for analyzing lattice properties in universal algebra. To demonstrate the robustness of our approach, we implemented different types of message passing layers, including graph convolution and GIN.

C Generalization results details

D Concept completeness and purity

Our experimental results demonstrate that interpretable GNNs produce concepts with high completeness and purity, which are standard quantitative metrics used to evaluate the quality of concept-based approaches. Specifically, our approach achieves up to X% completeness and Y purity, with an average score of Z and W, respectively. The lowest scores are obtained for the multilabel case, which is more challenging as models must learn a wider and more diverse set of concepts. Furthermore, the hierarchical structure of interpretable GNNs enables us to evaluate the quality of intermediate concepts layer by layer. This hierarchy provides insights into why we may need more layers, and

Table 1: Generalization performance of graph neural models in solving universal algebra’s tasks. Values represents the mean and the standard error of the mean of the area under the receiver operating curve (AUCROC, %).

	weak generalization			strong generalization		
	GNN	iGNN	HiGNN	GNN	iGNN	HiGNN
Distributive	99.80 ± 0.04	99.56 ± 0.12	99.45 ± 0.06	99.51 ± 0.20	99.44 ± 0.05	99.42 ± 0.04
Join Semi Distributive	99.49 ± 0.02	98.31 ± 0.15	98.28 ± 0.04	98.77 ± 0.15	97.50 ± 0.14	97.48 ± 0.14
Meet Semi Distributive	99.52 ± 0.04	98.19 ± 0.06	98.25 ± 0.08	98.90 ± 0.03	97.18 ± 0.14	96.89 ± 0.37
Modular	99.77 ± 0.02	99.18 ± 0.11	99.35 ± 0.09	99.32 ± 0.22	99.21 ± 0.14	99.11 ± 0.22
Semi Distributive	99.66 ± 0.03	98.57 ± 0.02	98.50 ± 0.06	99.19 ± 0.04	97.28 ± 0.48	96.88 ± 0.47
Multi Label	99.60 ± 0.02	96.32 ± 0.34	95.98 ± 0.50	98.62 ± 0.43	95.29 ± 0.55	95.27 ± 0.32

it can be used as a valuable tool to find the optimal setup and tune the size of the architecture. Additionally, it can also be used to compare the quality of concepts at different layers of the network. Overall, these results quantitatively assess and validate the high quality of the concepts learned by the interpretable GNNs, highlighting the effectiveness of this approach for learning and analyzing complex algebraic structures.

Table 2: Concept purity scores of graph neural models in solving universal algebra’s tasks.

	WEAK PURITY			STRONG PURITY		
	GNN	iGNN	HiGNN	GNN	iGNN	HiGNN
Distributive	3.30 ± 0.36	3.64 ± 0.30	3.09 ± 0.56	3.29 ± 0.38	4.00 ± 0.77	4.15 ± 0.67
Join Semi Distributive	2.38 ± 0.37	3.96 ± 0.51	3.74 ± 0.62	3.45 ± 0.34	3.98 ± 0.68	4.29 ± 0.61
Meet Semi Distributive	3.24 ± 0.63	3.55 ± 0.62	3.39 ± 0.29	3.36 ± 0.32	4.25 ± 0.39	4.97 ± 0.44
Modular	3.10 ± 0.35	3.50 ± 0.46	4.44 ± 0.56	3.14 ± 0.24	3.19 ± 1.01	4.25 ± 0.69
Semi Distributive	2.84 ± 0.51	3.70 ± 0.54	4.11 ± 0.46	3.70 ± 0.55	3.92 ± 0.28	4.08 ± 0.85

Table 3: Concept completeness scores of graph neural models in solving universal algebra’s tasks. Values represents the mean and the standard error of the mean of the area under the receiver operating curve (AUCROC, %).

	WEAK COMPLETENESS		STRONG COMPLETENESS	
	iGNN	HiGNN	iGNN	HiGNN
Distributive	77.30 ± 0.20	76.60 ± 2.35	78.19 ± 1.71	73.16 ± 3.63
Join Semi Distributive	85.20 ± 0.86	86.84 ± 0.37	81.08 ± 0.18	79.76 ± 0.38
Meet Semi Distributive	84.21 ± 0.68	84.34 ± 1.08	80.86 ± 1.32	79.68 ± 0.22
Modular	76.98 ± 0.28	73.77 ± 3.14	81.36 ± 0.70	77.61 ± 2.37
Semi Distributive	87.33 ± 1.16	85.62 ± 0.27	84.03 ± 0.89	82.26 ± 0.21