Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning
KR meets Machine Learning and Explanation

Unifying Approach to Uniform Expressivity of Graph Neural Networks

Huan Luo[1,2] , **Jonni Virtema**[1,2]
[1]School of Computing Science, University of Glasgow, UK
[2]School of Computer Science, University of Sheffield, UK
{huan.luo, jonni.virtema}@glasgow.ac.uk

Abstract

The expressive power of Graph Neural Networks (GNNs) is often analysed via correspondence to the Weisfeiler-Leman (WL) algorithm and fragments of first-order logic. Standard GNNs are limited to performing aggregation over immediate neighbourhoods or over global read-outs. To increase their expressivity, recent attempts have been made to incorporate substructural information (e.g., cycle counts and subgraph properties). In this paper, we formalise this architectural trend by introducing Template GNNs ($\mathcal{T}$-GNNs), a generalised framework where node features are updated by aggregating over valid template embeddings from a specified set of graph templates. We propose a corresponding logic, Graded template modal logic ($\mathrm{GML}(\mathcal{T})$), and generalised notions of template-based bisimulation and WL algorithm. We establish an equivalence between the expressive power of $\mathcal{T}$-GNNs and $\mathrm{GML}(\mathcal{T})$, and provide a unifying approach for analysing GNN expressivity: we show how standard AC-GNNs and its recent variants can be interpreted as instantiations of $\mathcal{T}$-GNNs.

1 Introduction

The proliferation of structured data such as graphs and relational structures, and rapid development in the area of neural network machine learning in the past two decades, have led to the development of bespoke machine learning architectures for structured data, most notably Graph Neural Networks (GNNs) (Scarselli et al. 2009). Various extensions and variations of this model have been introduced and studied—however, from a high level perspective, a GNN iteratively and synchronously updates a vector of numerical features in every node of a graph by combining the node's own feature vector with those of its neighbours (Gilmer et al. 2017). In this level of abstraction, the GNN model is a variation of the local model of distributed computing (Linial 1992).

Given an input graph, we want the GNN to output something meaningful. For example, this could be a property of the graph itself (e.g., decide whether the graph is Eulerian), a node property (e.g., inclusion to a dominating set), or link prediction (e.g., predict whether any two nodes are path-connected). In this paper, we focus on the task of node classification where in the end a Boolean-valued classification function is applied to the feature vector of each node. In this framework, GNNs express unary (i.e., node-selecting) queries on graphs, which in the area of graph learning are known as node classifiers.

Due to intimate connections between the capabilities of GNNs, expressivity of graph query languages, modal logics, and methods related to the graph isomorphism problem, a growing research community has revealed deep and precise connections between these seemingly disparate topics.

The (1-dimensional) Weisfeiler-Leman algorithm (Weisfeiler and Leman 1968)—a.k.a. colour refinement—is a well-known incomplete heuristic for deciding whether two graphs are isomorphic, which is known to have the same (non-uniform) expressivity as message passing GNNs (Morris et al. 2019; Xu et al. 2019).

In the modal logic community, de Rijke (2000) developed the notion of graded bisimulation—a relation between pointed labelled graphs—in order to characterise the expressivity of graded modal logic. It is not hard to see that the notions of graded bisimulation and colour refinement are two sides of the same coin; two points in a graph are graded (k-)bisimilar if and only if the points are in the same colour class after applying the 1-WL algorithm (k rounds).

Hella et al. (2015) discovered that there is a strong connection between the local model of distributed computing and graded modal logic. The paper established—utilising the notion of bisimulation—a one-to-one correspondence between local distributed algorithms and formulae of graded modal logic (in a non-uniform setting). Sato, Yamada, and Kashima (2019) further developed ideas from Hella et al. and applied them to the GNN setting.

The aforementioned characterisations of the expressivity of message passing models are all in some sense non-uniform. The connections to 1-WL relate to indistinguishability (with respect to any GNN of the given architecture), and the correspondence between local distributed algorithms and graded modal logic is formulated over degree bounded graphs. On the other hand, the uniform expressivity of a GNN architecture relates to studying the class of node classifiers that can be expressed by that GNN architecture (with respect to a full range of inputs that can be reasonably expected). The first characterisation of this kind was established by Barceló et al. (2020)—a logical classifier definable in first-order logic is captured by an aggregate-combine GNN (AC-GNN) if and only if it can be expressed in graded modal logic. They also established that each classifier definable in the two-variable fragment of first-order logic with counting quantifiers (C^2) can be captured by a simple homogeneous

aggregate-combine-readout GNN (ACR-GNN)—leaving the converse, relative to logical classifiers, open.

Since the seminal results of Barceló et al. (2020), a plethora of similar results have been shown for different GNN architectures. The connection between graded modal logic and AC-GNNs was further extended from logical classifiers to complete one-to-one correspondence by introducing mild forms of arithmetic to the logical side. Benedikt et al. (2024) utilised logics with so-called *Presburger quantifiers*, while Grohe (2024) utilised counting terms and built-in relations.

Cuenca Grau, Feng, and Wałęga (2026) gave a comprehensive picture of correspondences between AC(R)-GNNs and various modal logics—the main differentiator of their approach to that of Barceló et al. is the idea to replace the restriction to logical classifiers with the notion of bounded-GNNs (in the bounded setting, the ability of GNNs to differentiate multiplicities in a multiset is limited to some constant).

Hauke and Wałęga (2026) discovered that, actually, ACR-GNNs are strictly more expressive than C^2, even when restricted to logical classifiers, solving an open problem from Barceló et al. (2020). Their result indicates that, in general, when relating capabilities of GNNs to logics that are unable to do arithmetic, invariance under some bounded version of bisimulation is more appropriate restriction than restricting to logical classifiers. For instance, in the case of AC-GNNs, where a precise characterisation has been obtained in restriction to logical classifiers, invariance under graded bisimulation and FO-definability is equivalent to being invariant under bounded graded bisimulation (Otto 2019).

Provable limitations of the expressivity of AC-GNNs have led to various extensions, as very simple tasks such as deciding graph-reachability or cycle detection are already beyond their capabilities. Incorporating a form of recursion to GNN models allows properties such as graph-reachability to be expressible. Typically the non-uniform expressivity of GNN models of this type remains restricted to 1-WL, but their uniform expressivity is highly related to that of fixed-point logics such as the graded μ-calculus (Bollen et al. 2025; Pflueger, Tena Cucala, and Kostylev 2024; Ahvonen et al. 2024).

Two prominent variants for lifting GNN expressivity beyond 1-WL relate to enriching graph features with subgraph counts (Bouritsas et al. 2023; Bevilacqua et al. 2022; Frasca et al. 2022) or homomorphism pattern counts (Barceló et al. 2021; Jin et al. 2024)—a simple example here would be counting short cycles or paths, or homomorphisms to the complete graph of three vertices. Another branch of work considered weaker GNN models, e.g., Tena Cucala et al. (2023; 2024) related Max and Max-Sum GNNs to datalog.

Characterisations of the expressivity of various GNN architectures often use a similar recipe following the seminal result of Barceló et al. (2020): First, propose an extension or variation of the AC-GNN model, define a variant of the WL algorithm (or bisimulation) that corresponds to that GNN model, and a modal logic that has modalities corresponding to the extension. One then needs to show that the bisimulation yields an upper bound for the expressivity of the GNN, and that every GNN yields a parameter p such that p-bounded bisimulation is still an upper bound for its expressivity and has only finitely many p-bisimulation equivalence classes. Finally, one needs to show that every p-bisimulation equivalence class is definable in the newly defined logic, and that every logical formula can be simulated by a GNN—the latter is often done by computing the truth value of each subformula recursively in the elements of feature vectors.

While the recipe is simple to summarise, it does not mean that the generalisations are always easy to find or prove—sometimes only parts of the recipe can be applied. Some papers omit the logical counterpart completely and sometimes one gets a connection to logic only by allowing infinitary connectives—this is the case when one cannot prove that the number of relevant bisimulation equivalence classes is finite. Particular examples utilising parts of the above recipe include Bollen et al. (2025); Pflueger, Tena Cucala, and Kostylev (2024); Cuenca Grau, Feng, and Wałęga (2026); Soeteman and ten Cate (2025); Chen, Zhang, and Wang (2025).

Our contributions. We introduce an abstract general model of graph neural networks that utilises aggregation over *templates*. Templates are small patterns (i.e., graphs) that govern the form of message passing taking place in the GNN computation. In standard AC-GNNs, messages are passed from a direct neighbour to another—hence the *template* here is an edge. If on the other hand, in one communication round messages are only allowed to be passed within triangles, the template would be a triangle. Roughly speaking, aggregation in a template GNN with a template T is over the multiset of labelled templates that are obtained by collecting all embeddings of the template T into the graph (having the node that aggregates mapped to a specified node in the template). While our framework supports aggregation over multiple templates simultaneously, for a simpler instantiation we provide an example with a single directed triangle template.

Example 1. *Consider a board configuration in a Go-like game represented as a directed graph as in Figure 1, where nodes represent White or Black pieces, and arrows represent influence. When message passing is governed by a triangle template, node b is in three valid template embeddings and receives the following messages:* $\{\!\{$  $\}\!\}$. *Suppose aggregation functions are majority-rule operators, and that each node updates its colour according to the majority of received influences. In $\mathcal{T}$-GNNs there are two levels of aggregation: template aggregation and (multiset) aggregation. Node b receives one White message from* $\{\!\{$ $\}\!\}$ *and two Black messages from* $\{\!\{$  $\}\!\}$ *via template aggregation. A subsequent majority-rule aggregation hence yields a Black message, and node b updates to Black in the first round. Similarly, node c updates to Black in the first round. This leads to a Black majority message that node a receives in the second round and consequently updates to Black (Fig. 1, top). In standard message passing, node b and c each receive messages from their incoming edges:* $\{\!\{\circ, \circ, \circ, \bullet, \bullet\}\!\}$, *which aggregate to a White message. Both nodes remain White, leading to no change after two rounds of message passing (Fig. 1, bottom).*

After introducing $\mathcal{T}$-GNNs, we define the corresponding notions of $\mathcal{T}$-WL algorithm, graded $\mathcal{T}$-bisimulation, and graded template modal logic $\mathrm{GML}(\mathcal{T})$. We then establish that the expressivity of $\mathcal{T}$-GNNs is bounded by

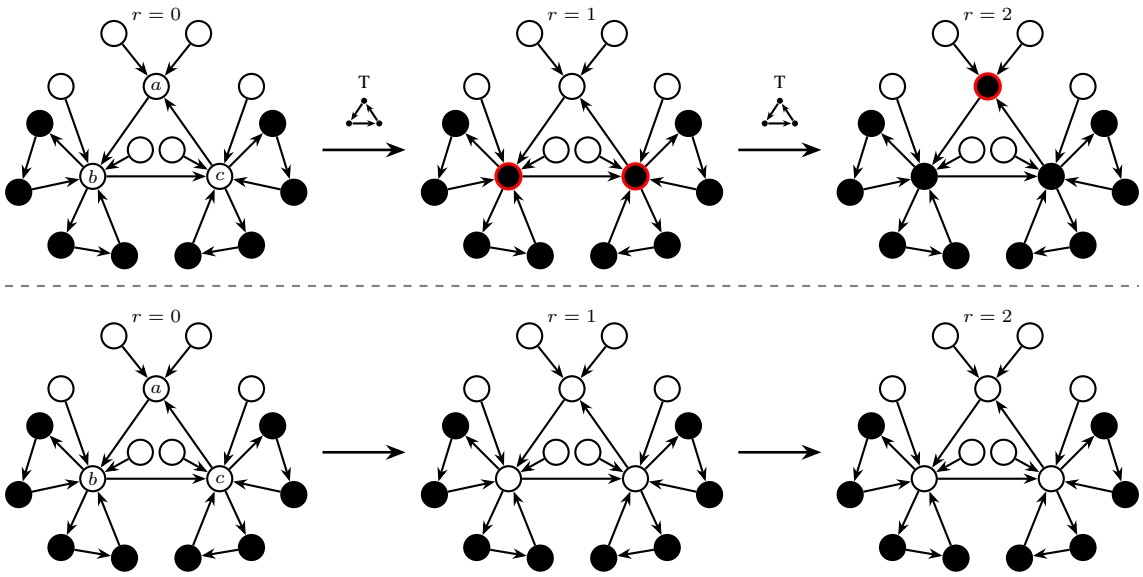

Figure 1: Illustration of two rounds of message passing in $\mathcal{T}$-GNNs (top) and AC-GNNs (bottom).

graded $\mathcal{T}$-bisimulation, which also bounds the expressivity of $\mathrm{GML}(\mathcal{T})$. Following Barceló et al. (2020), we show that every $\mathrm{GML}(\mathcal{T})$ formula can be simulated by a $\mathcal{T}$-GNN. Finally, we establish a one-to-one correspondence between the uniform expressivity of $\mathcal{T}$-GNNs and $\mathrm{GML}(\mathcal{T})$, in the bounded counting case introduced by Cuenca Grau, Feng, and Wałęga (2026) (here, in GNN aggregation, multiplicities in multisets are capped with some constant).

Our main contribution is the formalisation of a general GNN framework which can incorporate many of the modern paradigms for developing new expressive GNN architectures. Our main technical contribution, is a meta theorem yielding a family of theorems characterising the expressive power of GNNs. If one formalises their favourite GNN model as a template GNN, our results give direct definitions for the corresponding WL algorithm and logic that provably characterise the expressivity of the GNN model.

2 Preliminaries

Graphs. A *labelled directed graph* is a tuple $G = (V, E, \lambda)$, where V is a finite set of nodes, $E \subseteq V \times V$ is an edge relation, and $\lambda : V \to \mathbb{R}^d$ assigns to each node a vector of real numbers of dimension d, for a fixed $d \in \mathbb{N}$. (G, v) is a pointed graph given a graph G and a node $v \in V$. Two pointed graphs are isomorphic (written $(G, v) \cong (G', v')$) if there exists a bijection $f : V \to V'$ such that $f(v) = v'$, $\lambda(v) = \lambda'(f(v))$, for every $v \in V$, and $(v, u) \in E$ iff $(f(v), f(u)) \in E'$. For a tuple $t = (a_1, \ldots, a_n)$, t_i denotes its i-th element a_i. We identify assignments of type $\lambda : V \to \{0, 1\}^d$ with propositional assignments $\lambda' : V \to 2^{\{p_1, \ldots, p_d\}}$, in the obvious way. That is, $p_i \in \lambda'(v)$ if and only if $\lambda(v)_i = 1$. Such labelled directed graphs can be treated as Kripke structures with proposition set $\{p_1, \ldots, p_d\}$, and vice versa.

For a set S, we set S-Graphs to be the class of labelled directed graphs that are labelled with elements from S.

Graph classifiers and transformations. Let S and C be sets. A *node C-classifier* for S-Graphs is a function $f \colon S\text{-Graphs} \to C\text{-Graphs}$ that maintains the underlying graph structure (i.e., G and $f(G)$ may differ only on λ). If $C = \{0, 1\}$, this function is called a *Boolean node classifier*. If $C = S$, we call this function *S-graph transformation*.

GNN node classifiers. A standard aggregate-combine graph neural network (AC-GNN) consists of L AC layers and a classification function cls, for $L \in \mathbb{N}$. Intuitively, each GNN layer of input dimension d computes an $\mathbb{R}^d$-graph transformation, while the classification function computes a node C-classifier, for a finite set of classes C. The node classifier computed by the GNN is the composition of these functions. Formally, an AC layer of input dimension d is a pair $(\mathrm{agg}, \mathrm{comb})$, where agg is an aggregation function mapping multisets of vectors of dimension d to a vector of dimension d, and $\mathrm{comb} \colon \mathbb{R}^{2d} \to \mathbb{R}^d$ is a combination function mapping two vectors of dimension d to a vector of dimension d. At the l-th layer, the node label vector $\lambda^l(v)$ is updated via $comb\big(\lambda^{l-1}(v), \mathrm{agg}(\{\!\{\lambda^{l-1}(u)\}\!\}_{u \in N(v)})\big)$, where $N(v) = \{u \mid (v, u) \in E\}$ is the set of neighbours of v. Each node $v \in V$ is classified according to a classification function $\mathrm{cls} \colon \mathbb{R}^d \to C$ applied to the final node label $\lambda^L(v)$. An application $\mathcal{N}(G, v)$ of an AC-GNN $\mathcal{N}$ to a pointed labelled graph (G, v) is the value $\mathrm{cls}(\lambda^L(v))$.

Logical classifiers. Consider a finite set of propositions AP[1]. Formulae of basic modal logic (ML) are given by $\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Diamond\varphi$, where $p \in \mathrm{AP}$. Formulae are evaluated over pointed $\{0, 1\}^{|\mathrm{AP}|}$-labelled graphs (G, v) in the usual manner. Graded modal logic (GML) extends ML

[1]We restrict to finite AP to obtain a correspondence between AP and feature vectors of finite dimension $|\mathrm{AP}|$.

with graded modalities $\Diamond^{\geq c}$, for positive integers c, where

$$(G, v) \models \Diamond^{\geq c}\varphi \text{ iff } |\{u \mid (v, u) \in E \text{ and } (G, u) \models \varphi\}| \geq c.$$

A (Boolean) *GNN classifier $\mathcal{N}$ captures a logical classifier φ* if for every graph G and node v in G, it holds that $\mathcal{N}(G, v) = 1$ if and only if $(G, v) \models \varphi$.

Weisfeiler-Leman (WL) algorithm. The WL algorithm is an efficient heuristic for graph isomorphism (Weisfeiler and Leman 1968). Given a graph and a countable set of colours, the 1-dimensional WL (1-WL) is a colour refinement algorithm that updates node colouring according to the following rule: $C^l(v) = \mathrm{HASH}\big(C^{l-1}(v), \{\!\{C^{l-1}(u)\}\!\}_{u \in N(v)}\big)$, where HASH is assumed to be injective. The algorithm terminates when the colouring is stable or a pre-specified number of iterations is exceeded. Two graphs are WL indistinguishable if their final colour multisets match. It has been proven that GNNs and 1-WL have the same distinguishing power in the sense that a GNN can distinguish two nodes of a graph if and only if the colour refinement procedure assigns different colours to the nodes (Morris et al. 2019; Xu et al. 2019).

3 Template GNNs

We define our new model—template graph neural networks—and discuss how existing GNN models can be interpreted as template GNNs. First we introduce the notion of a template.

Definition 2 (Template)**.** *A* template *is a structure $T = (V, E^+, E^-, r)$ such that*

- *V is a finite set of vertices,*
- *$E^+ \subseteq V \times V$ is a set of edges, where $(u, v) \in E^+$ is interpreted to mean that there is an edge from u to v,*
- *$E^- \subseteq V \times V$ is a set of non-edges such that $E^+ \cap E^- = \emptyset$,*
- *$r \in V$ is a node called the root of the template.*

Note that, we do not presume that $E^+ \cup E^- = V \times V$.

A *labelled template* is a pair (T, λ), where $T = (V, E^+, E^-, r)$ is a template and $\lambda : V \to \mathbb{R}^n$ is a labelling.

Definition 3 (Template embedding)**.** *A template embedding of $T = (V, E^+, E^-, r)$ into a pointed graph $(G, w) = (V_G, E_G, w)$ is an injective homomorphism $f\colon V \to V_G$ where $f(r) = w$ and for all $u, v \in V$ it holds that*

1. *if $(u, v) \in E^+$ then $(f(u), f(v)) \in E_G$,*
2. *if $(u, v) \in E^-$ then $(f(u), f(v)) \notin E_G$.*

Note that, if $E^+ \cup E^- = V \times V$, a template embedding is simply an injective strong homomorphism from (V, E^+, r) to (G, w). We write $\mathrm{emb}(T, (G, w))$ for the set of all template embeddings of T into (G, w).

Definition 4 (Template isomorphism)**.** *Two labelled templates $(V_1, E_1^+, E_1^-, r_1, \lambda_1)$ and $(V_2, E_2^+, E_2^-, r_2, \lambda_2)$ are* template isomorphic *if there is a bijection $f\colon V_1 \to V_2$ such that*

1. *$f(r_1) = r_2$,*
2. *$(u, v) \in E_1^+$ if and only if $(f(u), f(v)) \in E_2^+$,*
3. *$(u, v) \in E_1^-$ if and only if $(f(u), f(v)) \in E_2^-$,*
4. *$\lambda_1(u) = \lambda_2(f(u))$, for every $u \in V_1 \setminus \{r_1\}$.*

A *(multiset) aggregation function* is any function that maps multisets of real numbers to a real number.

Definition 5 (Template aggregation function)**.** *Let $T = (V, E^+, E^-, r)$ be a template and $d \in \mathbb{N}$. A function $\mathrm{agg}_T\colon (T, \lambda) \to \mathbb{R}^d$ that maps labelled templates to $\mathbb{R}^d$ is a T-*aggregation function *if the function is invariant under template automorphisms. That is, $\mathrm{agg}_T(T, \lambda_1) = \mathrm{agg}_T(T, \lambda_2)$, whenever (T, λ_1) and (T, λ_2) are template isomorphic.*

Intuitively, a template GNN is a message passing GNN where messages are not passed via edges, but instead via template embeddings. For a fixed template T, the multiset of messages that a node v receives, is the multiset of labelled graphs obtained via template embeddings. The multiset of labelled graphs is then transformed into a multiset of feature vectors by the template aggregation function. This multiset of feature vectors is then combined with the feature vector of v as in standard AC-GNNs—via further aggregate and combine functions. The formal definition is given below.

Definition 6 (Unary Template GNN)**.** *Fix $L, d \in \mathbb{N}$ and a set of templates $\mathcal{T}$. A* unary L-layer $\mathcal{T}$-GNN *(with feature dimension d) is a tuple*

$$\mathcal{N} = (\{\mathrm{agg}^l_{T^l}\}_{l \leq L}, \{\mathrm{agg}^l\}_{l \leq L}, \{\mathrm{comb}^l\}_{l \leq L}, \mathrm{cls}),$$

where, for each layer $1 \leq l \leq L$, $T^l \in \mathcal{T}$ is a template, $\mathrm{agg}^l_{T^l}$ is a T^l-aggregation function, agg^l is an aggregation function, comb^l is a combination function, and $\mathrm{cls} : \mathbb{R}^d \to \{0, 1\}$ is a final Boolean classification function. At the l-th layer, the node feature vector (of dimension d) of a node v is updated by

$$\lambda^l(v) \coloneqq \mathrm{comb}^l\big(\lambda^{l-1}(v), \mathrm{agg}^l\{\!\{\mathrm{agg}^l_{T^l}(T^l, \lambda^{l-1}_f) \mid f \in \mathrm{emb}(T^l, (G, v))\}\!\}\big),$$

where $\lambda^{l-1}_f(u) := \lambda^{l-1}(f(u))$, for $u \in T^l$. Finally, each node v is classified by applying cls to the final node label $\lambda^L(v)$. When $\mathcal{T} = \{T\}$, we write T-GNN instead of $\mathcal{T}$-GNN. If $\mathcal{N}$ is a T-GNN for some $\mathcal{T}$, we call it a template GNN.

We also consider template GNNs that aggregate over multiple templates simultaneously. We call these $\mathcal{T}$-GNNs n-ary.

Definition 7 (n-ary Template GNN)**.** *Fix $L, d \in \mathbb{N}$ and a set of templates $\mathcal{T}$. An n-*ary L-layer $\mathcal{T}$-GNN *(with feature dimension d) is a tuple*

$$\mathcal{N} = (\{\mathrm{agg}^l_{T^l_j}\}_{\substack{l \leq L \\ j \leq n}}, \{\mathrm{agg}^l_j\}_{\substack{l \leq L, \\ j \leq n}}, \{\mathrm{comb}^l\}_{l \leq L}, \mathrm{cls}),$$

where, for each layer $1 \leq l \leq L$ and $1 \leq j \leq n$, $T^l_j \in \mathcal{T}$ is a template, $\mathrm{agg}^l_{T^l_j}$ is a T^l_j-aggregation function, agg^l_j is an aggregation function, comb^l is a combination function, and $\mathrm{cls} : \mathbb{R}^d \to \{0, 1\}$ is a final Boolean classification function. At the l-th layer, the feature vector of node v is updated by

$$\begin{aligned}\lambda^l(v) \coloneqq \mathrm{comb}^l\big(&\lambda^{l-1}(v), \\ &\mathrm{agg}^l_1\{\!\{\mathrm{agg}^l_{T^l_1}(T^l_1, \lambda^{l-1}_f) \mid f \in \mathrm{emb}(T^l_1, (G, v))\}\!\}, \\ &\vdots \\ &\mathrm{agg}^l_n\{\!\{\mathrm{agg}^l_{T^l_n}(T^l_n, \lambda^{l-1}_f) \mid f \in \mathrm{emb}(T^l_n, (G, v))\}\!\}\big),\end{aligned}$$

(a) T_1 (b) T_2

Figure 2: Templates T_1 and T_2 used to capture AC-GNNs and AC+-GNNs. Solid arrow represents an element of E^+, and dashed arrow represents an element of E^-.

where λ_f^{l-1} is defined as in the unary case.

For $c \geq 1$, we call a function f whose parameters are (tuples of) multisets *c-bounded* if $f(A) = f(A \upharpoonright c)$, where $A \upharpoonright c$ is obtained from A by changing multiplicities larger than c to c. We call a $\mathcal{T}$-GNN $\mathcal{N}$ *c-bounded*, if all of its outer aggregation functions[2] are c-bounded, and we call $\mathcal{N}$ *bounded*, if it is c-bounded for some c. For multisets A and B and $c \geq 1$, we write $A =_c B$ if $A \upharpoonright c = B \upharpoonright c$.

Families of GNNs interpreted as template GNNs. A standard AC-GNN can be interpreted as a unary template GNN with template T_1, as shown in Figure 2(a), which has vertex set $V = \{r_1, a\}$, edge set $E_1^+ = \{(r_1, a)\}$, and non-edge set $E_1^- = \emptyset$. The number of valid template embeddings at node v is the same as the number of neighbouring nodes of v. When agg_T projects to the feature vector of a and the outer aggregation function is the same as the one used in the AC-GNN, the update rules of node feature vectors for T-GNNs and AC-GNNs coincide.

Cuenca Grau, Feng, and Wałęga (2026) introduced the family of bounded GNNs, where aggregation (and readout) functions are restricted to c-bounded functions. The paper established that bounded GNNs with AC+ layers (defined below) have the same expressive power as the two variable fragment of first-order logic with counting quantifiers. For an AC+ layer, the node feature vectors $\lambda^l(v)$ is updated via

$$\mathrm{comb}\big(\lambda^{l-1}(v), \mathrm{agg}\{\!\{\lambda^{l-1}(u)\}\!\}_{u\in N(v)}, \mathrm{agg}\{\!\{\lambda^{l-1}(u)\}\!\}_{u\in \bar{N}(v)}\big),$$

where $\bar{N}(v) = \{u \mid (v,u) \notin E\} \setminus \{v\}$ is the set of non-neighbours of v excluding v itself. Bounded AC+-GNNs can be interpreted as a binary $\mathcal{T}$-GNN with $\mathcal{T} = \{T_1, T_2\}$ for each layer, where T_1 is the same as above, and T_2 with vertex set $V = \{r_2, a\}$, edge set $E_2^+ = \emptyset$, and non-edge set $E_2^- = \{(r_2, a)\}$, as shown in Figure 2(b). When agg_{T_1} and agg_{T_2} project to the feature vector of a, and the outer aggregation functions for T_1 and T_2 match the two aggregation functions used in AC+ layers, the update rules of node feature vectors for $\mathcal{T}$-GNNs and AC+-GNNs coincide.

Another line of work that can be directly interpreted as $\mathcal{T}$-GNNs is the ones that enrich the standard GNNs with local graph structural information. For instance, Chen, Zhang, and Wang (2025) proposed k-hop subgraph GNNs, where every node is updated by incorporating information extracted from subgraphs induced by the set of nodes reachable within k steps from the node. Specifically, the k-hop neighbourhood of a node v can be defined as $N_k(v) \coloneqq \{u \mid d(v,u) \leq k\}$ where $d(v,u)$ is the shortest path distance between v and u. We write G_v^k for the k-hop subgraph structure rooted at v. Instead of aggregating over immediate neighbourhood as in standard GNNs, k-hop subgraph GNNs aggregate over $(G_v^k, \{\!\{\lambda(u)^{(l-1)}\}\!\}_{u\in N_k(v)})$ when updating the node feature vectors. k-hop subgraph GNNs can be interpreted as $\mathcal{T}$-GNNs where $\mathcal{T}$ is the set of all rooted graphs of radius k. The radius of a template T is defined by $rd(T) \coloneqq \max_{v\in V} d(r,v)$, where $d(r,v)$ is the shortest E^+-path distance from r to v. If there is no E^+-path from r to v, we set $d(r,v) \coloneqq \infty$. Example 9 (in the next section) showcases how k-hop subgraph GNNs can be seen as template GNNs. See also Example 1 in page 2.

[2]Template aggregation functions are trivially c-bounded, since their input is a single labelled graph.

4 WL Algorithm and Bisimulation

We define the new notions of template WL algorithm and graded template bisimulation, and show that the notions coincide and bound the expressive power of template GNNs. We also discuss how existing WL variants can be interpreted in our framework.

Fix a countable set of colours C. A node colouring of a labelled graph $G = (V, E, \lambda)$ is a function $\mathrm{col}\colon V \to C$. Intuitively, template WL algorithm generalises the standard 1-WL algorithm by replacing, in the colour refinement rounds, the multiset of colours of neighbours of a given node by multiset of coloured graphs obtained via template embeddings (or tuple of multisets, when multiple templates are used).

Definition 8 ($\mathcal{T}$-WL algorithm)**.** *An L-round $\mathcal{T}$-WL algorithm takes as input a labelled graph $G = (V, E, \lambda)$, a finite set of templates $\mathcal{T} = \{T_1, \ldots, T_n\}$, and $L \in \mathbb{N}$. Initial node colours are given by node labels of the graph, $\mathrm{col}^0(v) \coloneqq \mathrm{HASH}(\lambda(v))$ for every $v \in V$. For $1 \leq l \leq L$, node colours are repeatedly refined at each round by*

$$\mathrm{col}^l(v) \coloneqq \mathrm{HASH}\big(\mathrm{col}^{l-1}(v), \{\!\{(T_1, \mathrm{col}_f^{l-1}) \mid f \in \mathrm{emb}(T_1, (G,v))\}\!\}, \ldots, \{\!\{(T_n, \mathrm{col}_f^{l-1}) \mid f \in \mathrm{emb}(T_n, (G,v))\}\!\}\big),$$

where $\mathrm{col}_f^{l-1}(u) \coloneqq \mathrm{col}^{l-1}(f(u))$, for each $u \in T_i$. The procedure terminates after L rounds, and $\mathrm{col}^L(v)$ is the final node colour. Here HASH *is assumed to be an injective function with co-domain C.*

When the $\mathcal{T}$-WL algorithm is run on a pair of graphs, it is run with the same injective HASH function.

The standard 1-WL algorithm updates a node's colour based on its own label and the multiset of its neighbours' colours. $\mathcal{T}$-WL with a single edge template, as shown in Figure 2(a), is equivalent to 1-WL. The update rule $\mathrm{col}^l(v) \coloneqq \mathrm{HASH}\big(\mathrm{col}^{l-1}(v), \{\!\{(T, \mathrm{col}_f^{l-1}) \mid f \in \mathrm{emb}(T, (G,v))\}\!\}$ is informationally equivalent to the standard 1-WL update rule: $\mathrm{col}^l(v) \coloneqq \mathrm{HASH}\big(\mathrm{col}^{l-1}(v), \{\!\{\mathrm{col}^{l-1}(u) \mid u \in N(v)\}\!\}\big)$.

To increase the expressive power of standard GNNs (from 1-WL), Chen, Zhang, and Wang (2025) proposed k-hop sub-

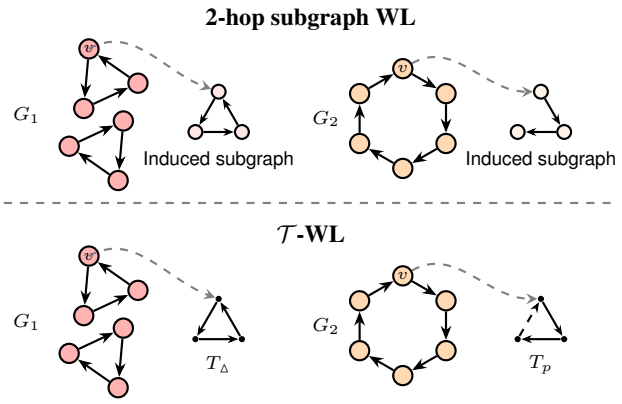

Figure 3: Top: 2-hop subgraph WL extracts subgraphs rooted at node v. Bottom: $\mathcal{T}$-WL arrives at the same colouring by matching node v against $\mathcal{T} = \{T_\triangle, T_p\}$.

graph GNNs and the corresponding k-hop subgraph WL algorithm. As discussed in the previous section, k-hop subgraph GNNs can be interpreted as $\mathcal{T}$-GNNs, when $\mathcal{T}$ is the set of all rooted graphs of radius k. Using the same set of templates, $\mathcal{T}$-WL algorithm corresponds to the k-hop subgraph WL algorithm of Chen, Zhang, and Wang. For illustrative purposes, the following example shows how 2-hop subgraph WL and $\mathcal{T}$-WL achieve the same result, on a pair of non-isomorphic graphs indistinguishable by the 1-WL algorithm.

Example 9. *Consider a triangle template $T_\triangle$ with vertex set $V = \{r, a, b\}$, edge set $E^+ = \{(r, a), (a, b), (b, r)\}$, and non-edge set $E^- = \emptyset$; and a path template T_p with vertex set $V = \{r, a, b\}$, edge set $E^+ = \{(r, a), (a, b)\}$, and non-edge set $E^- = \{(b, r)\}$. Figure 3 shows how 2-hop subgraph WL and $\mathcal{T}$-WL with $\mathcal{T} = \{T_\triangle, T_p\}$ distinguish a pair of non-isomorphic graphs, respectively.*

Next, we introduce the notion of graded template bisimulation and connect it to the $\mathcal{T}$-WL algorithm.

Definition 10 (Graded $\mathcal{T}$-bisimulation)**.** *Let $G = (V, E, \lambda)$ and $G' = (V', E', \lambda')$ be labelled graphs and $\mathcal{T}$ a set of templates. A relation $Z_0 \subseteq V \times V'$ is a graded 0-$\mathcal{T}$-bisimulation if for every $(v, v') \in Z_0$, $\lambda(v) = \lambda'(v')$. For $l \geq 1$, a relation $Z_l \subseteq V \times V'$ is a graded l-$\mathcal{T}$-bisimulation if there exists a graded $(l-1)$-$\mathcal{T}$-bisimulation Z_{l-1} such that for every $(v, v') \in Z_l$:*

1. *$(v, v') \in Z_{l-1}$,*
2. *for every $T \in \mathcal{T}$, and every $k \geq 1$:*
   - *for pairwise distinct $f_1, \ldots, f_k \in \mathrm{emb}(T, (G, v))$ there exists pairwise distinct $f'_1, \ldots, f'_k \in \mathrm{emb}(T, (G', v'))$ s.t. for all $1 \leq i \leq k$, $u \in T$, $(f_i(u), f'_i(u)) \in Z_{l-1}$,*
   - *for pairwise distinct $f'_1, \ldots, f'_k \in \mathrm{emb}(T, (G', v'))$ there exists pairwise distinct $f_1, \ldots, f_k \in \mathrm{emb}(T, (G, v))$ s.t. for all $1 \leq i \leq k$, $u \in T$, $(f_i(u), f'_i(u)) \in Z_{l-1}$.*

We say that (v, v') are graded l-$\mathcal{T}$-bisimilar*—and write $(G', v') \sim^l_{\mathcal{T}} (G, v)$—if there is a graded l-$\mathcal{T}$-bisimulation Z_l such that $(v, v') \in Z_l$. For a counting bound $c \geq 1$, we define l-c-$\mathcal{T}$-bisimulation by restricting $k \leq c$ in item 2, and write $(G', v') \sim^{l,c}_{\mathcal{T}} (G, v)$ when v and v' are l-c-$\mathcal{T}$-bisimilar.*

We obtain the notion of graded $\mathcal{T}$-bisimulation *(*graded $\mathcal{T}$-bisimilarity*, respectively) from the above definitions in the obvious way by replacing all occurrences of Z_0, Z_l, and Z_{l-1} by Z. For the cases of graded $\mathcal{T}$-bisimulation and graded l-$\mathcal{T}$-bisimulation, we may use the following alternative formulation of item 2:*

2'. for every template $T \in \mathcal{T}$, there exists a bijection $g : \mathrm{emb}(T, (G, v)) \to \mathrm{emb}(T, (G', v'))$ such that for all $f \in \mathrm{emb}(T, (G, v))$, for all $u \in T$, $(f(u), g(f)(u)) \in Z_{l-1}$.

$\mathcal{T}$-WL algorithm and graded $\mathcal{T}$-bisimulation are different sides of the same coin, as the following proposition shows.

Proposition 11. *Let $G = (V, E, \lambda)$ and $G' = (V', E', \lambda')$ be labelled graphs, $\mathcal{T}$ a finite set of templates, and $l \in \mathbb{N}$. Then $(G', v') \sim^l_{\mathcal{T}} (G, v)$ if and only if $\mathrm{col}^l(v) = \mathrm{col}^l(v')$.*

Proof. The proof is by induction on l. For the base case, by definition of initial colouring, $\mathrm{col}^0(v) = \mathrm{HASH}(\lambda(v))$ and $\mathrm{col}^0(v') = \mathrm{HASH}(\lambda'(v'))$. Since the hash function is injective, $\mathrm{col}^0(v) = \mathrm{col}^0(v')$ if and only if $\lambda(v) = \lambda'(v')$. By definition of graded 0-$\mathcal{T}$-bisimulation, $(v, v') \in Z_0$ if and only if $\lambda(v) = \lambda'(v')$. For the inductive step, assume the proposition holds for $l - 1$.

($\Leftarrow$): Since the hash function is injective, $\mathrm{col}^l(v) = \mathrm{col}^l(v')$ holds if and only if the arguments to the hash function are identical, as follows

1. $\mathrm{col}^{l-1}(v) = \mathrm{col}^{l-1}(v')$,
2. for every $T \in \mathcal{T}$, $\{\!\{(T, \mathrm{col}^{l-1}_f) \mid f \in \mathrm{emb}(T, (G, v))\}\!\} = \{\!\{(T, \mathrm{col}^{l-1}_{f'}) \mid f' \in \mathrm{emb}(T, (G', v'))\}\!\}$

By induction hypothesis (IH), item 1 implies $(v, v') \in Z_{l-1}$. The multisets are equal in item 2 implies for every $T \in \mathcal{T}$, there exists a bijection $g : \mathrm{emb}(T, (G, v)) \to \mathrm{emb}(T, (G', v'))$ such that for all $f \in \mathrm{emb}(T, (G, v))$, $(T, \mathrm{col}^{l-1}_f) = (T, \mathrm{col}^{l-1}_{g(f)})$, which implies $\mathrm{col}^{l-1}_f = \mathrm{col}^{l-1}_{g(f)}$, which means for all $u \in T$, $\mathrm{col}^{l-1}(f(u)) = \mathrm{col}^{l-1}(g(f)(u))$ by definition. By IH, it follows that for all $u \in T$, $(f(u), g(f)(u)) \in Z_{l-1}$. Since $\lambda(v) = \lambda'(v')$ is implied by initial colouring, $(v, v') \in Z_{l-1}$ is implied by 1., and the existence of bijection g for all $T \in \mathcal{T}$ is implied by item 2, the definition of graded l-$\mathcal{T}$-bisimulation is satisfied. Hence, $(v, v') \in Z_l$.

($\Rightarrow$): Assume $(v, v') \in Z_l$. Then $(v, v') \in Z_{l-1}$ by definition. By induction hypothesis, $(v, v') \in Z_{l-1}$ implies $\mathrm{col}^{l-1}(v) = \mathrm{col}^{l-1}(v')$, matching the first argument of the hash function. Since $(v, v') \in Z_l$, for every $T \in \mathcal{T}$, there exists a bijection $g : \mathrm{emb}(T, (G, v)) \to \mathrm{emb}(T, (G', v'))$ such that for all $f \in \mathrm{emb}$, for all $u \in T$, $(f(u), g(f)(u)) \in Z_{l-1}$ by definition. Applying induction hypothesis to all pairs $(f(u), g(f)(u)) \in Z_{l-1}$, we have $\mathrm{col}^{l-1}(f(u)) = \mathrm{col}^{l-1}(g(f)(u))$, which means $\mathrm{col}^{l-1}_f = \mathrm{col}^{l-1}_{g(f)}$. Since for every $T \in \mathcal{T}$ a bijection g exists mapping f to $g(f)$, $\{\!\{(T, \mathrm{col}^{l-1}_f) \mid f \in \mathrm{emb}(T, (G, v))\}\!\} = \{\!\{(T, \mathrm{col}^{l-1}_{f'}) \mid f' \in$

$\mathrm{emb}(T,(G',v'))\}\!\}$, matching the other arguments of the hash function. Therefore, $\mathrm{col}^l(v) = \mathrm{col}^l(v')$. □

It is not difficult too see that the above proof gives an analogous connection between the bounded variants of $\mathcal{T}$-WL algorithm and bounded $\mathcal{T}$-bisimulation. For $c \geq 1$, a c-bounded $\mathcal{T}$-WL algorithm is obtained from Definition 8 by stipulating that HASH cannot differentiate between multiplicities c from larger multiplicities (i.e., HASH is a c-bounded function). Hence, Proposition 11 can be reformulated to state that (v, v') are l-c-$\mathcal{T}$-bisimilar if and only if $\mathrm{col}^l(v) = \mathrm{col}^l(v')$ using a c-bounded HASH function.

Proposition 12. *Let G and G' be labelled graphs, $\mathcal{T}$ be a finite set of templates, $\mathcal{N}$ a c-bounded L-layer $\mathcal{T}$-GNN, and $l \leq L$. If $(v, v') \in V \times V'$ are l-c-$\mathcal{T}$-bisimilar, then they have the same feature vectors at the l-th layer of $\mathcal{N}$. The result remains true also for non-bounded $\mathcal{T}$-GNN, if l-c-$\mathcal{T}$-bisimilarity is replaced with graded l-$\mathcal{T}$-bisimilarity.*

Proof. The proof is by induction on l. For the base case, by definition of 0-c-$\mathcal{T}$-bisimulation, $(v, v') \in Z_0$ if and only if $\lambda(v) = \lambda(v')$. $\mathcal{T}$-GNN $\mathcal{N}$ initializes node feature vectors based on the initial node labels $\lambda(v)$ and $\lambda(v')$ of the graphs. For the inductive step, assume the statement holds for $l-1$.

Consider the update rule for a c-bounded $\mathcal{T}$-GNN classifier. Independent of choices of agg, agg_T, and comb functions, $\lambda^l(v) = \lambda^l(v')$ holds if

1. $\lambda^{l-1}(v) = \lambda^{l-1}(v')$,
2. for every $T \in \{T_1^l, \ldots, T_n^l\}$,

$$\{\!\{(T, \lambda_f^{l-1}) \mid f \in emb(T,(G,v))\}\!\} =_c \{\!\{(T, \lambda_{f'}^{l-1}) \mid f' \in emb(T,(G',v'))\}\!\}.$$

Assume $(v, v') \in Z_l$. Then $(v, v') \in Z_{l-1}$ by definition. By induction hypothesis $(v, v') \in Z_{l-1}$ implies $\lambda^{l-1}(v) = \lambda^{l-1}(v')$. By definition of l-c-$\mathcal{T}$-bisimulation, we have for every $T \in \mathcal{T}$, $k \leq c$, for pairwise distinct $f_1, \ldots, f_k \in \mathrm{emb}(T,(G,v))$ there exists pairwise distinct $f'_1, \ldots, f'_k \in \mathrm{emb}(T,(G',v'))$ s.t. for all $1 \leq i \leq k$, $u \in T$, $(f_i(u), f'_i(u)) \in Z_{l-1}$. Applying induction hypothesis to all pairs $(f_i(u), f'_i(u)) \in Z_{l-1}$, we have for all $1 \leq i \leq k$, $u \in T$, $\lambda^{l-1}(f_i(u)) = \lambda^{l-1}(f'_i(u))$, which means $\lambda_f^{l-1} = \lambda_{f'}^{l-1}$. The same arguments apply to the back condition symmetrically. The back and forth conditions ensure that the number of embeddings mapping to any tuple of equivalent classes in Z_{l-1} is preserved between (G, v) and (G', v') for $k \leq c$. Hence, for every $T \in \{T_1^l, \ldots, T_n^l\}$, $\{\!\{(T, \lambda_f^{l-1}) \mid f \in \mathrm{emb}(T,(G,v))\}\!\} =_c \{\!\{(T, \lambda_{f'}^{l-1}) \mid f' \in \mathrm{emb}(T,(G',v'))\}\!\}$. Therefore, $\lambda^l(v) = \lambda^l(v')$.

The proof for the unbounded case follows in verbatim, once all references to the counting bound c are removed. □

5 Graded Modal Logic and Template GNNs

We define our new logic—graded template modal logic $\mathrm{GML}(\mathcal{T})$, and prove the main technical result of the paper: for any finite set of templates $\mathcal{T}$, $\mathrm{GML}(\mathcal{T})$ captures the uniform expressivity of bounded counting $\mathcal{T}$-GNNs.

5.1 Graded Template Modal Logic

We define a logic that has a modality $\langle T\rangle^{\geq j}(\varphi_1, \ldots, \varphi_n)$ for each template T of cardinality $n+1$ and positive $j \in \mathbb{N}$. We henceforth stipulate that the domain of a template of cardinality $n+1$ is the set $[n+1] := \{0, 1, \ldots, n\}$ and that 0 is the root of the template.

GML($\mathcal{T}$). Let $\mathcal{T}$ be a finite set of templates. The syntax of $\mathrm{GML}(\mathcal{T})$ is given by

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle T\rangle^{\geq j}(\varphi_1, \varphi_2, \ldots, \varphi_{n_T}),$$

where $j \geq 1$ is a natural number, $T = (V, E^+, E^-, r) \in \mathcal{T}$ is a template, and $n_T = |V| - 1$.

The semantics extends that of GML with the following clause for formulae of the form $\langle T\rangle^{\geq j}(\varphi_1, \varphi_2, \ldots, \varphi_{n_T})$: $(G, v) \models \langle T\rangle^{\geq j}(\varphi_1, \varphi_2, \ldots, \varphi_{n_T})$ if and only if

$$|\{f \in \mathrm{emb}(T,(G,v)) \mid (G, f(i)) \models \varphi_i, \text{ for } 1 \leq i \leq n_T\}| \geq j.$$

The modal depth $\mathrm{md}(\varphi)$ of a $\mathrm{GML}(\mathcal{T})$ formula is defined as usual, with the following additional case:

$$\mathrm{md}(\langle T\rangle^{\geq j}(\varphi_1, \varphi_2, \ldots, \varphi_n)) := 1 + \max_{1 \leq i \leq n}(\mathrm{md}(\varphi_i)).$$

The depth of the syntactic tree $\mathrm{sd}(\varphi)$ of φ is defined similarly as $\mathrm{md}(\varphi)$, except that all logical connectives add 1 to the depth. We define the *counting bound* $\mathrm{cb}(\varphi)$ of a formula $\varphi \in \mathrm{GML}(\mathcal{T})$ to be the smallest natural number $c \in \mathbb{N}$ such that if $\langle T\rangle^{\geq j}$ occurs in φ, then $j \leq c$.

Proposition 13. *Every GML($\mathcal{T}$) formula of modal depth at most l and counting bound at most c is invariant under l-c-$\mathcal{T}$-bisimulation.*

Proof. The proof is by induction on the structure of φ. For the base case, let $\varphi = p$ be a proposition symbol. By definition, if (G, v) and (G', v') are l-c-$\mathcal{T}$-bisimilar, then $\lambda(v) = \lambda(v')$. Hence $(G, v) \models p \Leftrightarrow p \in \lambda(v) \Leftrightarrow p \in \lambda'(v') \Leftrightarrow (G', v') \models p$.

For the inductive step, let φ be a formula with $\mathrm{md}(\varphi) \leq l$ and $\mathrm{cb} \leq c$. The Boolean cases follow immediately from the induction hypothesis. Consider the case where $\varphi = \langle T\rangle^{\geq j}(\varphi_1, \ldots, \varphi_n)$. Since $\mathrm{md}(\varphi) \leq l$, we have $\mathrm{md}(\varphi_i) \leq l-1$, for $1 \leq i \leq n$. By definition of counting bound, $j \leq c$.

Let S be the set

$$\{f \in \mathrm{emb}(T,(G,v)) \mid (G, f(i)) \models \varphi_i, \text{ for } 1 \leq i \leq n\},$$

and S' be the corresponding set

$$\{f' \in \mathrm{emb}(T,(G',v')) \mid (G', f'(i)) \models \varphi_i, \text{ for } 1 \leq i \leq n\}.$$

By the forth condition in the definition of l-c-$\mathcal{T}$-bisimulation, for pairwise distinct $f_1, \ldots, f_j \in \mathrm{emb}(T,(G,v))$ there exists pairwise distinct $f'_1, \ldots, f'_j \in \mathrm{emb}(T,(G',v'))$ s.t. for all $1 \leq m \leq j$, $u \in T$, $(f_m(u), f'_m(u)) \in Z_{l-1}$. Symmetrically, by the back condition in the definition of l-c-$\mathcal{T}$-bisimulation, we have for pairwise distinct $f'_1, \ldots, f'_j \in \mathrm{emb}(T,(G',v'))$ there exists pairwise distinct $f_1, \ldots, f_j \in \mathrm{emb}(T,(G,v))$ s.t. for all $1 \leq m \leq j$, $u \in T$, $(f_m(u), f'_m(u)) \in Z_{l-1}$. Since $(f_m(u), f'_m(u)) \in Z_{l-1}$ and $\mathrm{md}(\varphi_i) \leq l-1$, by induction hypothesis, we have $(G, f(i)) \models \varphi_i \Leftrightarrow (G', f'(i)) \models \varphi_i$, for $1 \leq i \leq n$. Hence, $(G, v) \models \varphi \Leftrightarrow |S| \geq j \Leftrightarrow |S'| \geq j \Leftrightarrow (G', v') \models \varphi$. □

Next, we define two variants of the characteristic formulae $\chi^l_{(G,v)}$ and $\chi^{l,c}_{(G,v)}$ for GML($\mathcal{T}$), where $l, c \in \mathbb{N}$ and (G, v) is a finite pointed graph. The goal of these formulae is to satisfy the following two properties:

$$(G', v') \models \chi^l_{(G,v)} \Leftrightarrow (G', v') \sim^l_{\mathcal{T}} (G, v),$$
$$(G', v') \models \chi^{l,c}_{(G,v)} \Leftrightarrow (G', v') \sim^{l,c}_{\mathcal{T}} (G, v).$$

Since we are interested in connections to neural networks, we restrict considerations to finite graphs. In the case of infinite graphs, $\chi^l_{(G,v)}$ would not always be a finite formula.

For a template T of cardinality $n+1$, an n-tuple of GML formulae $\vec{\varphi}$, and a pointed graph (G, v), we define $S^{(G,v)}_{T,\vec{\varphi}}$ to be the following set of template embeddings:

$$\{f \in \mathrm{emb}(T, (G, v)) \mid (G, f(i)) \models \varphi_i \text{ for } 1 \leq i \leq n\}.$$

Definition 14 (Characteristic formulae for GML)**.** *Let $\mathcal{T}$ be a finite set of templates, and (G, v) a pointed labelled graph. We define the* l-characteristic formula $\chi^l_{(G,v)}$ *of (G, v) by induction on $l \in \mathbb{N}$ as follows:*

$$\chi^0_{(G,v)} \coloneqq \bigwedge\{p \mid p \in \lambda(v)\} \wedge \bigwedge\{\neg p \mid p \notin \lambda(v)\}.$$

For $l \geq 1$, the characteristic formula $\chi^l_{(G,v)}$ is defined as

$$\chi^l_{(G,v)} \coloneqq \chi^{l-1}_{(G,v)} \wedge \bigwedge_{T \in \mathcal{T}} \Big(\bigwedge_{f \in \mathrm{emb}(T,(G,v))} \langle T \rangle^{\geq k}(\varphi^f_1, \ldots, \varphi^f_n) \text{ where } k = |S^{(G,v)}_{T,\vec{\varphi}}| \wedge \neg \langle T \rangle^{\geq |S^{(G,v)}_{T,\vec{\top}}|+1}(\vec{\top}) \Big),$$

where $\vec{\varphi} = (\varphi^f_1, \ldots, \varphi^f_n) = (\chi^{l-1}_{(G,f(1))}, \ldots, \chi^{l-1}_{(G,f(n))})$ and $\vec{\top} = (\top, \ldots, \top)$.

Next, we define the bounded variants of l-characteristic formulae. We first require the following proposition, whose proof is standard. We say that an equivalence relation $\sim$ has a *finite index*, if it has finitely many equivalence classes. We write $[\sim]$ for the set of all $\sim$ equivalence classes.

Proposition 15. *For every $l, c \in \mathbb{N}$ and a finite set of templates $\mathcal{T}$, the relation $\sim^{l,c}_{\mathcal{T}}$ has a finite index.*

For the next definition, we need a representative for each equivalence class $C \in [\sim^{l,c}_{\mathcal{T}}]$; we may pick, e.g., the first in the alphabetical ordering obtained via some encoding. We write $(G, v) \in [\sim^{l,c}_{\mathcal{T}}]$ when ranging over such representatives.

Definition 16 (Bounded characteristic formulae for GML)**.** *Let $c \geq 1$ be a natural number, $\mathcal{T}$ a finite set of templates, and (G, v) a pointed labelled graph. We define the* l-c-characteristic formula $\chi^{l,c}_{(G,v)}$ *of (G, v) by induction on $l \in \mathbb{N}$ as follows:*

$$\chi^{0,c}_{(G,v)} \coloneqq \bigwedge\{p \mid p \in \lambda(v)\} \wedge \bigwedge\{\neg p \mid p \notin \lambda(v)\}.$$

For $l \geq 1$, the characteristic formula $\chi^{l,c}_{(G,v)}$ is defined as

$$\chi^{l,c}_{(G,v)} \coloneqq \chi^{l-1,c}_{(G,v)} \wedge \bigwedge_{T \in \mathcal{T}} \Big(\bigwedge_{f \in \mathrm{emb}(T,(G,v))} \langle T \rangle^{\geq k}(\vec{\varphi}) \quad \text{where } k = \min\{c, |S^{(G,v)}_{T,\vec{\varphi}}|\} \wedge \bigwedge_{\substack{\{(G_i,v_i) \in [\sim^{l-1,c}_{\mathcal{T}}]\}_{1 \leq i \leq n} \\ |S^{(G,v)}_{T,\vec{\psi}}|+1 \leq c}} \neg \langle T \rangle^{\geq |S^{(G,v)}_{T,\vec{\psi}}|+1}(\psi_1, \ldots, \psi_n) \Big),$$

where $\vec{\varphi} = (\varphi^f_1, \ldots, \varphi^f_n) = (\chi^{l-1,c}_{(G,f(1))}, \ldots, \chi^{l-1,c}_{(G,f(n))})$, $\vec{\psi} = (\psi_1, \ldots, \psi_n) = (\chi^{l-1,c}_{(G_1,v_1)}, \ldots, \chi^{l-1,c}_{(G_n,v_n)})$.

The following proposition follows directly from the construction of the characteristic formulae.

Proposition 17. *Every pointed graph (G, v) satisfies its own l-c-$\mathcal{T}$-characteristic formula $\chi^{l,c}_{(G,v)}$, for every $l, c \in \mathbb{N}$.*

We are now ready to prove the desired property that a characteristic formula of a pointed graph characterises the corresponding bisimulation equivalence class.

Proposition 18. *Let $c \geq 1$ and $l \in \mathbb{N}$ be natural numbers and $\mathcal{T}$ a finite set of templates. Then $(G', v') \models \chi^{l,c}_{(G,v)} \Leftrightarrow (G', v') \sim^{l,c}_{\mathcal{T}} (G, v)$. Hence, every $\sim^{l,c}_{\mathcal{T}}$ equivalence class is definable by a GML($\mathcal{T}$) formula of modal depth l and counting bound c.*

Proof. ($\Leftarrow$): Assume $(G', v') \sim^{l,c}_{\mathcal{T}} (G, v)$. By Proposition 17, $(G, v) \models \chi^{l,c}_{(G,v)}$. By Proposition 13, therefore, $(G', v') \models \chi^{l,c}_{(G,v)}$.

($\Rightarrow$): The proof proceeds by induction on l. For the base case, by definition of $\chi^{0,c}_{(G,v)}$, $(G', v') \models \chi^{0,c}_{(G,v)}$ iff $\lambda(v') = \lambda(v)$, matching the definition of $(G', v') \sim^{0,c}_{\mathcal{T}} (G, v)$. For the inductive step, assume the proposition holds for $l-1$.

Assume $(G', v') \models \chi^{l,c}_{(G,v)}$. By definition of $\chi^{l,c}_{(G,v)}$, $(G', v') \models \chi^{l-1,c}_{(G,v)}$. By induction hypothesis, $(G', v') \sim^{l-1,c}_{\mathcal{T}} (G, v)$, matching the first condition in the definition of $\sim^{l,c}_{\mathcal{T}}$. For any $T \in \mathcal{T}$, let $\vec{C} = (C_1, \ldots, C_n)$ be any tuple of representatives of the equivalence classes of $\sim^{l-1,c}_{\mathcal{T}}$, and $\vec{\psi}_{\vec{C}}$ be the tuple of corresponding characteristic formulae. Let $n = |S^{(G,v)}_{T,\vec{\psi}_{\vec{C}}}|$ and $m = |S^{(G',v')}_{T,\vec{\psi}_{\vec{C}}}|$. In the case of $n \geq c$, the second conjunct in the definition of $\chi^{l,c}_{(G,v)}$ requires $\langle T \rangle^{\geq c}\vec{\psi}$, which implies $m \geq c$. The third conjunct does not apply as $n+1 \leq c$ is not met. Hence, $\min(c, n) = c = \min(c, m)$. In the case of $n < c$, the second conjunct requires $\langle T \rangle^{\geq n}\vec{\psi}$, which implies $m \geq n$. As $n+1 \leq c$, the third conjunct applies and requires $\neg\langle T \rangle^{\geq n+1}\vec{\psi}$, which implies $m \leq n$. Taken together, $m = n$. Hence, $\min(c, n) = n = m = \min(c, m)$.

Let $f_1, \ldots, f_k \in \mathrm{emb}(T, (G, v))$ be pairwise distinct embeddings, $k \leq c$. Group them by the characteristic formulae $\vec{\psi}_{\vec{C}}$ satisfied by the image of template nodes. For any specific group of size $k' \leq k \leq c$, $k' \leq n$. Hence, $k' \leq \min(c, n) = \min(c, m)$. This implies G' contains at least k' distinct embeddings where the image of template

nodes satisfy $\vec{\psi}_{\vec{C}}$. By induction hypothesis, nodes satisfying the same characteristic formulae are $(l-1)$-c-T-bisimilar. Symmetric arguments apply to the back condition in the definition of $\sim_{\mathcal{T}}^{l,c}$. Therefore, $(G', v') \sim_{\mathcal{T}}^{l,c} (G, v)$. □

Now, since there are only finitely many bounded bisimulation classes for any fixed parameters, we obtain that every l-c-$\mathcal{T}$ invariant class of pointed graphs is definable in $\mathrm{GML}(\mathcal{T})$.

Proposition 19. *Every l-c-$\mathcal{T}$ invariant class of pointed graphs is definable by $\mathrm{GML}(\mathcal{T})$ formula of modal depth l and counting bound c.*

Proof. Every l-c-$\mathcal{T}$ invariant class of pointed graphs is a union of $\sim_{\mathcal{T}}^{l,c}$ equivalence classes. Every $\sim_{\mathcal{T}}^{l,c}$ equivalence class is definable by a l-c-characteristic formula by Proposition 18, and there is a finite number of such equivalence classes by Proposition 15. Therefore, every l-c-$\mathcal{T}$ invariant class of pointed graphs is definable by a finite disjunction of l-c-characteristic formulae, one for each equivalence class. □

5.2 From GNNs to logic and back

We are now ready to prove our main technical results connecting the uniform expressivity of $\mathcal{T}$-GNNs with the expressivity of $\mathrm{GML}(\mathcal{T})$. Combining our results on $\mathcal{T}$-GNN invariance under bounded graded bisimulation and Proposition 19 allow us to establish the following result of the uniform expressivity of $\mathcal{T}$-GNNs. Recall that a logical classifier φ captures a (Boolean) GNN classifier $\mathcal{N}$ if for every graph G and node v in G, it holds that $\mathcal{N}(G, v) = 1$ if and only if $(G, v) \models \varphi$.

Theorem 20. *Let $\mathcal{T}$ be a finite set of templates and $\mathcal{N}$ a c-bounded $\mathcal{T}$-GNN with l layers. Then there exists a $\varphi \in \mathrm{GML}(\mathcal{T})$ of modal depth l and counting bound c that captures $\mathcal{N}$.*

Proof. The class of pointed graphs that a c-bounded $\mathcal{T}$-GNN $\mathcal{N}$ with l layers accepts is invariant under l-c-$\mathcal{T}$-bisimulation by Proposition 12. Every such class of pointed graphs is definable by a $\mathrm{GML}(\mathcal{T})$ formula of modal depth l and counting bound c by Proposition 19. □

Theorem 21. *Let $\mathcal{T}$ be a finite set of templates, $\varphi \in \mathrm{GML}(\mathcal{T})$, $c_{max} = \mathrm{cb}(\varphi)$ and $l = sd(\varphi)$. Then there exists a (bounded) $\mathcal{T}$-GNN $\mathcal{N}$ with l layers that captures φ.*

Proof. Let $sub(\varphi) = (\varphi_1, \ldots, \varphi_d)$ be the set of all subformulae of φ, such that if φ_k is a subformula of φ_l, then $k \leq l$. In particular, $\varphi = \varphi_d$.

We construct a (bounded) l layer $\mathcal{T}$-GNN $\mathcal{N}$ of dimension d. The final classification function is $\mathrm{cls}(\lambda^l(v)_d)$, where $\lambda^l(v)_d$ is the d-th component of $\lambda^l(v)$.

Let $J \subseteq \{1, \ldots, d\}$ be the set of indices s.t. for all $k \in J$, φ_k is the modal formula of the form $\langle T \rangle^{\geq c}(\varphi_1, \varphi_2, \ldots, \varphi_n)$. Let $m = |J|$ be the number of modal subformulae in $sub(\varphi)$. We define a bijection $\iota : \{1, \ldots, m\} \to J$ s.t. for each $j \in \{1, \ldots, m\}$, let the corresponding subformula be: $\varphi_{\iota(j)} = \langle T^{(j)} \rangle^{\geq c_j}(\psi_{j,1}, \ldots, \psi_{j,n_j})$.

Layer 0 initializes all feature vectors to $\mathbb{R}^d$ to represent truth values of propositions, such that if φ_k is a proposition, then $\lambda^0(v)_k = 1$ iff $(G, v) \models \varphi_k$, all other entries are set to 0. Layers $1, \ldots, l$ are homogeneous, with activation function being truncated ReLU, defined as $\sigma(x) = \min(\max(0, x), 1)$. Aggregation function is the max-n-sum where n is the counting bound of φ. For each layer l and each $j \in \{1, \ldots, m\}$, assign the template $T_j^l = T^{(j)}$, corresponding to $\varphi_{\iota(j)} = \langle T^{(j)} \rangle^{\geq c_j}(\psi_{j,1}, \ldots, \psi_{j,n_j})$, let $id(j, i)$ be the index of the subformula $\psi_{j,i}$ in $sub(\varphi)$. The template aggregation function is defined as follows:

$$\mathrm{agg}^l_{T^{(j)}}(T^{(j)}, \lambda_f^{l-1}) \coloneqq \sigma\left(\sum_{i=1}^{n_j} \lambda^{l-1}(f(i))_{id(j,i)} - n_j + 1\right) \quad (1)$$

The combination function is defined as

$$\mathrm{comb}(\mathbf{x}, z_1, \ldots, z_m) = \sigma(\mathbf{x}\mathbf{C} + \mathbf{z}\mathbf{A} + \mathbf{b}),$$

where $\mathbf{x} \in \mathbb{R}^d$ is $\lambda^{l-1}(v)$, $\mathbf{z} = (z_1, \ldots, z_m) \in \mathbb{R}^m$ are obtained from template aggregations, and matrices $\mathbf{C} \in \mathbb{R}^{d \times d}, \mathbf{A} \in \mathbb{R}^{m \times d}, \mathbf{b} \in \mathbb{R}^d$ are defined based on $sub(\varphi)$ as follows:

1. If φ_k is a proposition, then $C_{kk} = 1$,
2. If $\varphi_k = \neg\varphi_p$, then $C_{pk} = -1$ and $b_k = 1$,
3. If $\varphi_k = \varphi_p \wedge \varphi_q$, then $C_{pk} = C_{qk} = 1$, and $b_k = -1$,
4. If $\varphi_k = \varphi_{\iota(j)}$ corresponds to a modal formula of the form $\langle T^{(j)} \rangle^{\geq c_j}(\psi_{j,1}, \ldots, \psi_{j,n_j})$, then $A_{jk} = 1$ and $b_k = -c_j + 1$,

and all other entries are 0.

Next we show by induction on the structure of subformula φ_k that for every graph G, every node v, and any layer $l \geq \mathrm{sd}(\varphi_k)$, $\lambda^l(v)_k = 1$ iff $(G, v) \models \varphi_k$.

For the base case, if φ_k is a proposition, $\mathrm{sd}(\varphi_k) = 0$. By design of layer 0, $\lambda^0(v)_k = 1$ iff $(G, v) \models \varphi_k$, and $\lambda^l(v)_k = 0$ otherwise. As $C_{kk} = 1$ preserves identity, $\lambda^l(v)_k = 1$ iff $(G, v) \models \varphi_k$ for all $l \geq 0$.

For the inductive step, assume the statement holds for all subformulae of syntactic depth less than φ_k.

- Case 1: $\varphi_k = \neg\varphi_p$. By construction, $C_{pk} = -1$ and $b_k = 1$, we have $\lambda^l(v)_k = \sigma(-\lambda^{l-1}(v)_p + 1)$. By induction hypothesis, $\lambda^{l-1}(v)_p = 1$ iff $(G, v) \models \varphi_p$, then $\lambda^l(v)_k = \sigma(0) = 0$. Similarly, $\lambda^{l-1}(v)_p = 0$ iff $(G, v) \not\models \varphi_p$, then $\lambda^l(v)_k = \sigma(1) = 1$. Hence, $\lambda^l(v)_k$ captures $\neg\varphi_p$.
- Case 2: $\varphi_k = \varphi_p \wedge \varphi_q$. By construction, $C_{pk} = C_{qk} = 1$ and $b_k = -1$, we have $\lambda^l(v)_k = \sigma(\lambda^{l-1}(v)_p + \lambda^{l-1}(v)_q - 1)$. By induction hypothesis, $\lambda^{l-1}(v)_p = 1$ iff $(G, v) \models \varphi_p$, and $\lambda^{l-1}(v)_q = 1$ iff $(G, v) \models \varphi_q$. Hence, $\lambda^l(v)_k = 1$ iff $(G, v) \models \varphi_p \wedge \varphi_q$, and $\lambda^l(v)_k = 0$ otherwise.
- Case 3: $\varphi_k = \varphi_{\iota(j)} = \langle T^{(j)} \rangle^{\geq c_j}(\psi_{j,1}, \ldots, \psi_{j,n_j})$. Consider first the template aggregation function. By inductive hypothesis, $\lambda^{l-1}(f(i))_{id(j,i)} = 1$ iff $(G, f(i)) \models \psi_i$. The sum in (1) equals to n iff all subformulae $(\psi_{j,1}, \ldots, \psi_{j,n_j})$ are satisfied, in which case $\sigma(n - n + 1) = 1$. The sum is $\leq n - 1$ otherwise, in which case σ returns 0. Next consider the max-n-sum aggregation function bounded by c_{max}. Let N be the number of valid embeddings. By construction, $A_{j,k} = 1$ and $b_k = -c + 1$, we have $\lambda^l(v)_k = \sigma(\min(c_{max}, N) - c + 1)$. Since $c \leq c_{max}$,

if $N \geq c$, then $\min(c_{max}, N) \geq c$, $\lambda^l(v)_k = 1$. If $N < c$, $\lambda^l(v)_k = 0$. Hence, $\lambda^l(v)_k = 1$ iff $N \geq c$ iff $(G, v) \models \varphi_k$, and $\lambda^l(v)_k = 0$ otherwise. □

By combining Theorems 20 and 21 we obtain that Boolean bounded $\mathcal{T}$-GNN node classifiers are exactly those that are definable in $\mathrm{GML}(\mathcal{T})$.

6 Conclusions and Future Work

We introduced template GNNs as a framework to study the (uniform) expressivity of different graph neural networks in a unified manner. In addition, we introduced the accompanying notions of $\mathcal{T}$-WL algorithm, graded $\mathcal{T}$-bisimulation, and graded template modal logic $\mathrm{GML}(\mathcal{T})$. We showed how various existing approaches to extend the expressivity of GNNs utilising diverse subgraph information can be formalised in our framework.

The main technical result of our paper is a metatheorem stating that, for any finite set of templates $\mathcal{T}$, $\mathrm{GML}(\mathcal{T})$ captures the uniform expressivity of bounded counting $\mathcal{T}$-GNNs. Several existing characterisations, such as the ones by Barceló et al. (2020) and Cuenca Grau, Feng, and Wałęga (2026) can be seen as special cases of our metatheorem. Similarly, our metatheorem is directly applicable to the k-hop subgraph GNNs of Chen, Zhang, and Wang (2025) and their version of the 1-WL algorithm.

A recent result by Hauke and Wałęga (2026) (discussed in the introduction) implies that, in general, our logical characterisation does not transfer to the case of unbounded $\mathcal{T}$-GNNs and $\mathrm{GML}(\mathcal{T})$, even when restricted to logical classifiers definable in first-order logic. However, since trivially $\mathcal{T}$-GNNs and bounded $\mathcal{T}$-GNNs have the same separation power—on any given graph, $\mathcal{T}$-GNNs aggregate over multisets of bounded cardinality—two graphs are separable by a $\mathcal{T}$-GNN if they are separable by the $\mathcal{T}$-WL algorithm, or equivalently, by a $\mathrm{GML}(\mathcal{T})$ formula.

Open questions and future work. We conclude by discussing directions for future research.

- Our framework is closely connected to the local graph parameter enriched GNNs of Barceló et al. (2021). Their $\mathcal{F}$-MPNNs and $\mathcal{F}$-WL algorithm obtain graph pattern information as our model does, but restricts to standard graph neighbour message passing. What is the precise relationship between these models?
- Zhang et al. (2024) relates the expressivity of various MPNN architectures to the class of graphs for which the architecture is capable to count homomorphisms. Can their approach be used to pinpoint the expressivity of particular classes of $\mathcal{T}$-GNNs.
- Can we extend our framework to cover the Hierarchical Ego Graph Neural Networks (Soeteman and ten Cate 2025) by some kind of hybrid extension of our logic.
- Can we extend our logical characterisations to cover unbounded $\mathcal{T}$-GNNs by adding similar counting features to our logic as in Benedikt et al. (2024); Grohe (2024).
- Can we extend our results to cover recurrent neural networks using a form of μ-calculus as in Bollen et al. (2025).

AI Declaration

The authors have not employed any Generative AI tools.

References

Ahvonen, V.; Heiman, D.; Kuusisto, A.; and Lutz, C. 2024. Logical characterizations of recurrent graph neural networks with reals and floats. In Globerson, A.; Mackey, L.; Belgrave, D.; Fan, A.; Paquet, U.; Tomczak, J.; and Zhan, C., eds., *Advances in Neural Information Processing Systems*, volume 37, 104205–104249.

Barceló, P.; Kostylev, E. V.; Monet, M.; Pérez, J.; Reutter, J.; and Silva, J.-P. 2020. The logical expressiveness of graph neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Barceló, P.; Geerts, F.; Reutter, J. L.; and Ryschkov, M. 2021. Graph neural networks with local graph parameters. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y. N.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 25280–25293.

Benedikt, M.; Lu, C.; Motik, B.; and Tan, T. 2024. Decidability of graph neural networks via logical characterizations. In Bringmann, K.; Grohe, M.; Puppis, G.; and Svensson, O., eds., *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, Tallinn, Estonia, July 8-12, 2024*, volume 297 of *LIPIcs*, 127:1–127:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Bevilacqua, B.; Frasca, F.; Lim, D.; Srinivasan, B.; Cai, C.; Balamurugan, G.; Bronstein, M. M.; and Maron, H. 2022. Equivariant subgraph aggregation networks. In *10th International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Bollen, J.; Van den Bussche, J.; Vansummeren, S.; and Virtema, J. 2025. Halting recurrent gnns and the graded μ-calculus. In *Proceedings of the 22nd International Conference on Principles of Knowledge Representation and Reasoning*, 175–184.

Bouritsas, G.; Frasca, F.; Zafeiriou, S.; and Bronstein, M. M. 2023. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45(1):657–668.

Chen, Z.; Zhang, Q.; and Wang, R. 2025. On the expressive power of subgraph graph neural networks for graphs with bounded cycles. *arXiv preprint arXiv:2502.03703*.

Cuenca Grau, B.; Feng, E.; and Wałęga, P. A. 2026. The correspondence between bounded graph neural networks and fragments of first-order logic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 19135–19142.

de Rijke, M. 2000. A note on graded modal logic. *Studia Logica* 64(2):271–283.

Frasca, F.; Bevilacqua, B.; Bronstein, M. M.; and Maron, H. 2022. Understanding and extending subgraph gnns by rethinking their symmetries. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Ad-*

vances in Neural Information Processing Systems, volume 35, 31376–31390.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, 1263–1272.

Grohe, M. 2024. The descriptive complexity of graph neural networks. *TheoretiCS* 3:Article 25, 1–93.

Hauke, S. P., and Wałęga, P. A. 2026. Aggregate-combine-readout gnns are more expressive than logic C^2. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 21594–21601.

Hella, L.; Järvisalo, M.; Kuusisto, A.; Laurinharju, J.; Lempiäinen, T.; Luosto, K.; Suomela, J.; and Virtema, J. 2015. Weak models of distributed computing, with connections to modal logic. *Distributed Computing* 28(1):31–53.

Jin, E.; Bronstein, M. M.; Ceylan, İ. İ.; and Lanzinger, M. 2024. Homomorphism counts for graph neural networks: All about that basis. In *Proceedings of the 41st International Conference on Machine Learning*, 22075–22098.

Linial, N. 1992. Locality in distributed graph algorithms. *SIAM Journal on Computing* 21(1):193–201.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4602–4609.

Otto, M. 2019. Graded modal logic and counting bisimulation. *arXiv preprint arXiv:1910.00039*.

Pflueger, M.; Tena Cucala, D.; and Kostylev, E. V. 2024. Recurrent graph neural networks and their connections to bisimulation and logic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 14608–14616.

Sato, R.; Yamada, M.; and Kashima, H. 2019. Approximation ratios of graph neural networks for combinatorial problems. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32, 4083–4092.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20(1):61–80.

Soeteman, A., and ten Cate, B. 2025. Logical expressiveness of graph neural networks with hierarchical node individualization. *arXiv preprint arXiv:2506.13911*. To appear in NeurIPS 2025.

Tena Cucala, D. J., and Cuenca Grau, B. 2024. Bridging max graph neural networks and datalog with negation. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning*, 950–961.

Tena Cucala, D. J.; Cuenca Grau, B.; Motik, B.; and Kostylev, E. V. 2023. On the correspondence between monotonic max-sum gnns and datalog. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*, 658–667.

Weisfeiler, B., and Leman, A. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya* 2(9):12–16.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhang, B.; Gai, J.; Du, Y.; Ye, Q.; He, D.; and Wang, L. 2024. Beyond weisfeiler-lehman: A quantitative framework for gnn expressiveness. *arXiv preprint arXiv: 2401.08514*.