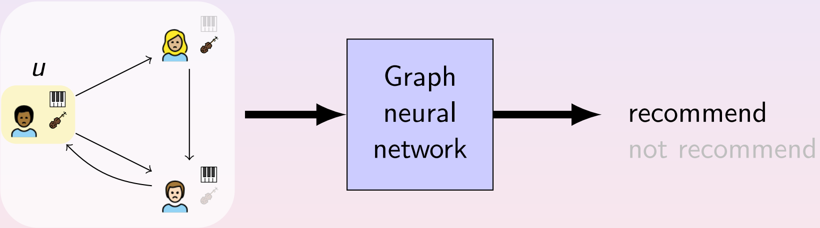
A Logic for Reasoning about Aggregate-Combine Graph Neural Networks

Pierre Nunn, Marco Sälzer, François Schwarzentruber, Nicolas Troquard

Univ Rennes, IRISA, CNRS / University of Kassel / GSSI

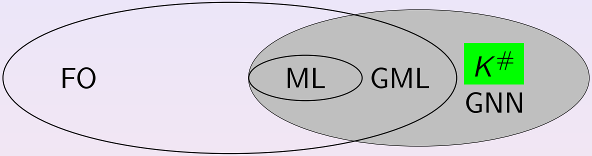
CNIA 2024

Motivation



labeled pointed graph

Contribution: logic $K^\#$ that corresponds to GNNs



FO = first-order logic

GML = graded modal logic

Theorem (Barceló et al. 2020)

- *Any ~~GML~~ $K^\#$ formula has an equivalent GNN*
- *Any GNN ~~FO-expressible~~ has an equivalent ~~GML~~ $K^\#$ formula*

+ transformations are poly-time

Contribution: methodology to solve verification problems

Let $\mathcal{A}$ be a GNN.
$[[\mathcal{A}]] :=$ set of *labeled pointed graphs* recommended by the GNN $\mathcal{A}$

Let φ be a logical formula. e.g. in modal logic, or in $K^{\#}$
$[[\varphi]] =$ set of *labeled pointed graphs* satisfying the property φ

Definition (verification problems)

Given a formula φ, a GNN $\mathcal{A}$, decide whether:
$[[\mathcal{A}]] = [[\varphi]]$ $[[\mathcal{A}]] \subseteq [[\varphi]]$ $[[\varphi]] \subseteq [[\mathcal{A}]]$ $[[\mathcal{A}]] \cap [[\varphi]] \neq \emptyset$

Contribution: methodology to solve verification problems

Theorem

The satisfiability problem of $K^{\#}$ is PSPACE-complete.

By poly-time reduction to the logic in [Demri and Lugiez 2010]

Corollary

The verification problems for logic $K^{\#}$ are also PSPACE-complete.

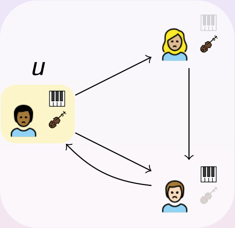
Outline

1. Our logic $K^{\#}$
2. Graph neural networks
3. Discussions

Example of a formula of $K^{\#}$

Consider formula φ:

$$\mathit{pianist} \ \wedge \ [\#\mathit{violinist} + \#(\#\mathit{pianist} \geq 1) \leq 3]$$



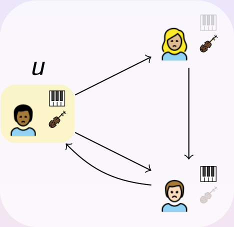
The semantics $[[\varphi]]$ is the set of pointed graphs G, u such that:

u is pianist and [the number of u-friends that are violinists + the number of u-friends (having at least one pianist friend) is ≤ 3]

Outline

1. Our logic $K^{\#}$
2. Graph neural networks
3. Discussions

A labeled pointed graph



Definition (global state)

$$x : V \to \mathbb{R}^d$$

Example

$x_0($  $) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ $x_0($  $) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ $x_0($  $) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

A GNN = an algorithm for recommending pointed graphs

input: a labeled pointed graph (G, u)
output: recommend or not
function $\mathcal{A}(G, u)$

x_0 := initial global state of G

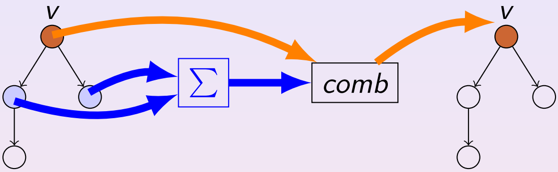
$x_1 := layer_1(G, x_0)$

$\vdots$

$x_L := layer_L(G, x_{L-1})$

return $\begin{cases} \text{recommend if } cls(x_L(u)) \geq 0 \\ \text{not recommend otherwise} \end{cases}$

Each layer



Each $layer_i$ is a function of the form:

input: a labeled graph G, a global state x output: global state

function $layer(G, x)$

return the mapping $v \mapsto comb(x(v), \sum\{\{x(w) \mid vEw\}\})$

with $comb : (\alpha, \beta) \mapsto \sigma(A\alpha + B\beta + b)$
where σ is a component-wise activation function
A, B are matrices, b a vector

Outline

1. Our logic $K^\#$
2. Graph neural networks
3. Discussions

Limitations

For having the correspondence, we suppose:

- the activation function is **truncated-ReLU**
- coefficients in the matrices A, B and vectors b are **integers**

For technical reasons:

- Formulas are represented as circuits (directed acyclic graphs)
- Logic $K^{\#}$ also contain a construction 1_{φ}

Other Recent Works

[Benedikt and al. ICALP 2024]

- Similar to our work
- Generalizes some of our results
- Their logic is based on guarded FO
- Algorithm is given directly on GNN, not with the logic correspondence
- No verification problem addresses

[Yang, Chiang, https://arxiv.org/abs/2404.04393, 2024]

A linear temporal variant of $K^{\#}$ for transformers

- Thanks to Stéphane Demri
- Thanks to the existence of M1 research project at ENS Rennes
- Thanks to Marco Sälzer and Nicolas Troquard

and

Thank you!