On Logical Characterisation of Graph Neural Network

Debanjan Dutta

April 17, 2024

Table of Contents

Introduction

Models of Computation

Model of AC-GNN

A graph neural network (GNN) is a dynamics map $\mathcal{F} : \mathbb{Q}^{n\times d} \to \mathbb{Q}^{n\times d}$.

Model of AC-GNN

A graph neural network (GNN) is a dynamics map $\mathcal{F} : \mathbb{Q}^{n\times d} \to \mathbb{Q}^{n\times d}$. Given a graph G, a vertex v and $L \in \mathbb{N}$; the **A**ggregate **C**ombine-**G**raph **N**eural **N**etwork (AC-GNN) model $\mathcal{A}$ performs a partitioning task (similar to vertex colouring) on v as follows:

$$\boldsymbol{x}_v^{(l)} = \mathrm{COM}^{(l)}\left(\boldsymbol{x}_v^{(l-1)}, \mathrm{AGG}^{(l)}\left(\{\!\{\boldsymbol{x}_u^{(l-1)} \mid \boldsymbol{x}_u \in \mathcal{N}(v)\}\!\}\right)\right) \quad l \in [L] \text{ and } \boldsymbol{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)} \quad (1)$$

where

$$\begin{aligned}
\mathrm{AGG}^{(l)} &: \text{ is any aggregation function} \\
\mathrm{COM}^{(l)}(\boldsymbol{v}, \boldsymbol{u}) &: f(\boldsymbol{A}^{(l)}\boldsymbol{v} + \boldsymbol{C}^{(l)}\boldsymbol{u} + \boldsymbol{b}) \\
f(x) &= \min(\max(x, 0), 1) \\
\mathcal{F}^l = \mathcal{F}(\boldsymbol{X}_G^{(l)}) &= \{\!\{\boldsymbol{x}_v^{(l+1)} \mid \text{ for all vertex } v \text{ in } G \}\!\}.
\end{aligned}$$

Finally, given a function CLS the node classification of a vertex v in G is defined by

$$\mathcal{A}(G, v) = \mathrm{CLS}(\boldsymbol{x}_v^{(L)}) \in \{\mathtt{true}, \mathtt{false}\}.$$

Model of AC-GNN

A graph neural network (GNN) is a dynamics map $\mathcal{F} : \mathbb{Q}^{n\times d} \to \mathbb{Q}^{n\times d}$. Given a graph G, a vertex v and $L \in \mathbb{N}$; the **A**ggregate **C**ombine-**G**raph **N**eural **N**etwork (AC-GNN) model $\mathcal{A}$ performs a partitioning task (similar to vertex colouring) on v as follows:

$$\mathbf{x}_v^{(l)} = \mathrm{COM}^{(l)}\left(\boldsymbol{x}_v^{(l-1)}, \mathrm{AGG}^{(l)}\left(\{\!\{\boldsymbol{x}_u^{(l-1)} \mid \boldsymbol{x}_u \in \mathcal{N}(v)\}\!\}\right)\right) \quad l \in [L] \text{ and } \mathbf{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)} \quad (1)$$

where

$$\begin{aligned} \mathrm{AGG}^{(l)} &: \text{ is any aggregation function} \\ \mathrm{COM}^{(l)}(\boldsymbol{v}, \boldsymbol{u}) &: f(\boldsymbol{A}^{(l)}\boldsymbol{v} + \boldsymbol{C}^{(l)}\boldsymbol{u} + \boldsymbol{b}) \\ f(x) &= \min(\max(x, 0), 1) \\ \mathcal{F}^l &= \mathcal{F}(\boldsymbol{X}_G^{(l)}) = \{\!\{\boldsymbol{x}_v^{(l+1)} \mid \text{ for all vertex } v \text{ in } G\}\!\}. \end{aligned}$$

Finally, given a function CLS the node classification of a vertex v in G is defined by

$$\mathcal{A}(G, v) = \mathrm{CLS}(\boldsymbol{x}_v^{(L)}) \in \{\mathit{true}, \mathit{false}\}.$$

Some Properties:

1. Parameters of $\mathcal{A}$: $\mathrm{COM}^{(l)}, \mathrm{AGG}^{(l)}$ and CLS.
2. Homogeneity: if $\mathrm{COM}^{(l)}$ $\mathrm{AGG}^{(l)}$ are identical through out all $l \in [L]$

Models of Computation

Model of AC-GNN

A graph neural network (GNN) is a dynamics map $\mathcal{F} : \mathbb{Q}^{n\times d} \to \mathbb{Q}^{n\times d}$. Given a graph G, a vertex v and $L \in \mathbb{N}$; the **A**ggregate **C**ombine-**G**raph **N**eural **N**etwork (AC-GNN) model $\mathcal{A}$ performs a partitioning task (similar to vertex colouring) on v as follows:

$$\boldsymbol{x}_v^{(l)} = \mathrm{COM}^{(l)}\left(\boldsymbol{x}_v^{(l-1)}, \mathrm{AGG}^{(l)}\left(\{\!\{\boldsymbol{x}_u^{(l-1)} \mid \boldsymbol{x}_u \in \mathcal{N}(v)\}\!\}\right)\right) \quad l \in [L] \text{ and } \boldsymbol{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)} \tag{1}$$

where

$$\begin{aligned}
\mathrm{AGG}^{(l)} &: \text{ is any aggregation function} \\
\mathrm{COM}^{(l)}(\boldsymbol{v}, \boldsymbol{u}) &: f(\boldsymbol{A}^{(l)}\boldsymbol{v} + \boldsymbol{C}^{(l)}\boldsymbol{u} + \boldsymbol{b}) \\
f(x) &= \min(\max(x, 0), 1) \\
\mathcal{F}^l &= \mathcal{F}(\boldsymbol{X}_G^{(l)}) = \{\!\{\boldsymbol{x}_v^{(l+1)} \mid \text{ for all vertex } v \text{ in } G\}\!\}.
\end{aligned}$$

Finally, given a function CLS the node classification of a vertex v in G is defined by

$$\mathcal{A}(G, v) = \mathrm{CLS}(\boldsymbol{x}_v^{(L)}) \in \{\mathtt{true}, \mathtt{false}\}.$$

Some Properties:

1. Parameters of $\mathcal{A}$: $\mathrm{COM}^{(l)}, \mathrm{AGG}^{(l)}$ and CLS.
2. Homogeneity: if $\mathrm{COM}^{(l)}$ $\mathrm{AGG}^{(l)}$ are identical through out all $l \in [L]$

Some Variants:

1. Model ACR-GNN: $\boldsymbol{x}_v^{(l)} = \mathrm{COM}^{(l)}\left(\boldsymbol{x}_v^{(l-1)}, \mathrm{AGG}^{(l)}\left(\{\!\{\boldsymbol{x}_u^{(l-1)} \mid \boldsymbol{x}_u \in \mathcal{N}(v)\}\!\}\right), \mathrm{ReadOut}^{(l)}\left(\{\!\{\boldsymbol{x}_u^{(l-1)} \mid \boldsymbol{x}_u \in G.V\}\!\}\right)\right)$
2. AC-FR-GNN: ReadOut applied only to the last layer.

A few Words regarding AC-GNN

On a drive to exploit the node classification with respect to our paradigm of logic we introduce the definition as follows:

Example of a logical classifier:

$$\varphi(v) = \mathrm{Red}(v) \wedge \exists u \, \mathrm{Green}(u)$$

Definition (AC-GNN as Node-Colour Classifier)

A GNN classifier $\mathcal{A}$ captures a logical classifier $\varphi(x)$ if for every graph G and node v in G, it holds that $\mathcal{A}(G, v) = \mathtt{true}$ if and only if $(G, v) \models \varphi$.

A few Words regarding AC-GNN

On a drive to exploit the node classification with respect to our paradigm of logic we introduce the definition as follows:

Example of a logical classifier:

$$\varphi(v) = \mathrm{Red}(v) \wedge \exists u\, \mathrm{Green}(u)$$

Definition (AC-GNN as Node-Colour Classifier)

A GNN classifier $\mathcal{A}$ captures a logical classifier $\varphi(x)$ if for every graph G and node v in G, it holds that $\mathcal{A}(G, v) = \mathtt{true}$ if and only if $(G, v) \models \varphi$.

Observations:

1. The model suffers from the flow of information of *local* aggregations that cannot travel further than a fixed distance $L + 1$.
2. *Global* information will only be captured when ReadOut function is incorporated.

Applications:

1. In the realm of geometric deep learning GNN are used vastly under tasks that fall under node classification.
2. There are also some tasks that fall under Edge prediction: Citation prediction, Probable Co-Author prediction.
3. Hybrid tasks.

Representational Power of AC-GNN

$\mathbf{FOC_2}$ Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

$\mathbf{FOC_2}$ Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists u(\neg E(u, x) \wedge \mathrm{Green}(u))$$
- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists y(\neg E(x, y) \wedge \mathrm{Blue}(y))$$
$$\wedge \exists u \exists v(\neg E(u, x)\, \mathrm{Green}(u) \wedge (E(v, x) \wedge \mathrm{Blue}(v) \wedge \neg(u = v))$$

$\mathbf{FOC_2}$ Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

- $$\varphi(x) = \text{Red}(x) \wedge \exists u(\neg E(u, x) \wedge \text{Green}(u))$$
- $$\varphi(x) = \text{Red}(x) \wedge \exists y(\neg E(x, y) \wedge \text{Blue}(y))$$
$$\wedge \exists u \exists v(\neg E(u, x)\, \text{Green}(u) \wedge (E(v, x) \wedge \text{Blue}(v) \wedge \neg(u = v))$$
- Can we reduce the number of variables?

FOC_2 Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists u(\neg E(u, x) \wedge \mathrm{Green}(u))$$
- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists y(\neg E(x, y) \wedge \mathrm{Blue}(y))$$
  $$\wedge \exists u \exists v(\neg E(u, x)\, \mathrm{Green}(u) \wedge (E(v, x) \wedge \mathrm{Blue}(v) \wedge \neg(u = v))$$
- Can we reduce the number of variables?
  Replace v by y.

FOC_2 Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists u(\neg E(u, x) \wedge \mathrm{Green}(u))$$
- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists y(\neg E(x, y) \wedge \mathrm{Blue}(y))$$ $$\wedge \exists u \exists v(\neg E(u, x)\, \mathrm{Green}(u) \wedge (E(v, x) \wedge \mathrm{Blue}(v) \wedge \neg(u = v))$$
- Can we reduce the number of variables?
  Replace v by y.

Definition (FO_2 Logic)

An FO_2 logic family is subpart of first order logic that involves only two variables.

$\mathbf{FOC_2}$ Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists u(\neg E(u,x) \wedge \mathrm{Green}(u))$$
- $$\begin{gathered}\varphi(x) = \mathrm{Red}(x) \wedge \exists y(\neg E(x,y) \wedge \mathrm{Blue}(y)) \\ \wedge \exists u \exists v(\neg E(u,x)\, \mathrm{Green}(u) \wedge (E(v,x) \wedge \mathrm{Blue}(v) \wedge \neg(u=v))\end{gathered}$$
- Can we reduce the number of variables?
  Replace v by y.

Definition ($\mathbf{FO_2}$ Logic)

An FO_2 logic family is subpart of first order logic that involves only two variables.

Still we are interested in computing in such logic. So consider an operator $\exists^{\geq n}$ implying existence of at least $n \geq 1$ *distinct* variable satisfying the succeeding formula.

FOC_2 Logic Family

Recall that, AC-GNN are shy to fetch the global information in a graph.

- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists u(\neg E(u, x) \wedge \mathrm{Green}(u))$$
- $$\varphi(x) = \mathrm{Red}(x) \wedge \exists y(\neg E(x, y) \wedge \mathrm{Blue}(y))$$ $$\wedge \exists u \exists v(\neg E(u, x)\, \mathrm{Green}(u) \wedge (E(v, x) \wedge \mathrm{Blue}(v) \wedge \neg(u = v))$$
- Can we reduce the number of variables?
  Replace v by y.

Definition (FO_2 Logic)

An FO_2 logic family is subpart of first order logic that involves only two variables.

Still we are interested in computing in such logic. So consider an operator $\exists^{\geq n}$ implying existence of at least $n \geq 1$ *distinct* variable satisfying the succeeding formula.

Definition (FOC_2 Logic)

The segment of FO_2 logic that involves such a *counting quantifier* is known as FOC_2 logic class

$$\gamma(x) = \mathrm{Green}(x) \wedge \exists^{\geq 2} y\, \mathrm{Blue}(y).$$

$\mathbf{FOC_2}$ and AC-GNN

Proposition

There is an FOC_2 classifier that is not captured by any AC-GNN.

$\mathbf{FOC_2}$ and AC-GNN

Proposition

There is an FOC_2 classifier that is not captured by any AC-GNN.

Let us consider an FO_2 formula $\delta(v) = \mathrm{Red}(v) \wedge \exists u \, \mathrm{Blue}(u)$.

- Suppose AC-GNN $\mathcal{A}$ with number of layers $L \in \mathbb{N}$ can satisfy $\delta(v)$ for any graph G.
- Take $G = P_{L+2}$ that is a path graph of length $L + 2$ such that

$$v \to u_1 \to \ldots \to u_L \to u_{L+1}$$

- $\mathcal{A}(P_{L+2}, v)$ can capture information up to node u_L.
- Although $(P_{L+2}, v) \models \delta$, $\mathcal{A}(P_{L+2}, v) = \mathtt{false}$, no matter what we choose for CLS, AGG or COM.

FOC_2 and AC-GNN

Proposition

There is an FOC_2 classifier that is not captured by any AC-GNN.

Let us consider an FO_2 formula $\delta(v) = \mathrm{Red}(v) \wedge \exists u \, \mathrm{Blue}(u)$.

- Suppose AC-GNN $\mathcal{A}$ with number of layers $L \in \mathbb{N}$ can satisfy $\delta(v)$ for any graph G.
- Take $G = P_{L+2}$ that is a path graph of length $L + 2$ such that

$$v \to u_1 \to \ldots \to u_L \to u_{L+1}$$

- $\mathcal{A}(P_{L+2}, v)$ can capture information up to node u_L.
- Although $(P_{L+2}, v) \models \delta$, $\mathcal{A}(P_{L+2}, v) = \texttt{false}$, no matter what we choose for CLS, AGG or COM.

Let us loosen the the constraint and let $\mathcal{A}$ run for $|G.E|$ times.

- Suppose AC-GNN $\mathcal{A}$ with number of layers $L = |G.E|$ can satisfy $\delta(v)$ for any graph G.
- Consider a disconnected graph with two components $G = (\{v, v_1, \ldots, v_n\} \cup \{u\}, E)$.
- Even if $(G, v) \models \delta$, $\mathcal{A}(G, v) = \texttt{flase}$.

FOC_2 and AC-GNN

Proposition

There is an FOC_2 classifier that is not captured by any AC-GNN.

Let us consider an FO_2 formula $\delta(v) = \mathrm{Red}(v) \wedge \exists u\, \mathrm{Blue}(u)$.

- Suppose AC-GNN $\mathcal{A}$ with number of layers $L \in \mathbb{N}$ can satisfy $\delta(v)$ for any graph G.
- Take $G = P_{L+2}$ that is a path graph of length $L+2$ such that

$$v \to u_1 \to \ldots \to u_L \to u_{L+1}$$

- $\mathcal{A}(P_{L+2}, v)$ can capture information up to node u_L.
- Although $(P_{L+2}, v) \models \delta$, $\mathcal{A}(P_{L+2}, v) = \texttt{false}$, no matter what we choose for CLS, AGG or COM.

Let us loosen the the constraint and let $\mathcal{A}$ run for $|G.E|$ times.

- Suppose AC-GNN $\mathcal{A}$ with number of layers $L = |G.E|$ can satisfy $\delta(v)$ for any graph G.
- Consider a disconnected graph with two components $G = (\{v, v_1, \ldots, v_n\} \cup \{u\}, E)$.
- Even if $(G, v) \models \delta$, $\mathcal{A}(G, v) = \texttt{flase}$.

Clearly, AC-GNN neither satisfy FO_2 formula nor FOC_2.

1. So what kind of logic family can AC-GNN capture?
2. And what model should capture FOC_2?

Graded Modal Logic

Definition

If every bound variable in a FOC_2 formula involves an edge constraints with it we will call them *graded* modal logic. Thus a graded modal formula φ is of form:

$$\varphi(x) : \mathrm{Col}(x) \mid \neg\varphi(x) \mid \varphi(x) \wedge \psi(x) \mid \exists^{\geq n} y(E(x,y) \wedge \varphi(y)). \tag{2}$$

Proposition

Each graded modal logic classifier is captured by a simple homogeneous AC-GNN.

Graded Modal Logic

Definition

If every bound variable in a FOC_2 formula involves an edge constraints with it we will call them *graded* modal logic. Thus a graded modal formula φ is of form:

$$\varphi(x) : \mathrm{Col}(x) \mid \neg\varphi(x) \mid \varphi(x) \wedge \psi(x) \mid \exists^{\geq n} y(E(x, y) \wedge \varphi(y)). \tag{2}$$

Proposition

Each graded modal logic classifier is captured by a simple homogeneous AC-GNN.

- $\mathcal{A}_\varphi$ satisfies graded modal formula φ.
- Suppose the graded modal logic formula φ involves L sub-formulas $(\varphi_1, \varphi_2, \ldots, \varphi_L)$ such that if φ_k is a sub-formula of φ_l then $k < l$.

Graded Modal Logic

Definition

If every bound variable in a FOC_2 formula involves an edge constraints with it we will call them *graded* modal logic. Thus a graded modal formula φ is of form:

$$\varphi(x) : \mathrm{Col}(x) \mid \neg\varphi(x) \mid \varphi(x) \wedge \psi(x) \mid \exists^{\geq n} y(E(x, y) \wedge \varphi(y)). \tag{2}$$

Proposition

Each graded modal logic classifier is captured by a simple homogeneous AC-GNN.

- $\mathcal{A}_\varphi$ satisfies graded modal formula φ.
- Suppose the graded modal logic formula φ involves L sub-formulas $(\varphi_1, \varphi_2, \ldots, \varphi_L)$ such that if φ_k is a sub-formula of φ_l then $k < l$.

- Embedding dimension of each node $\boldsymbol{x}_v$ to be L.
- The number of iterations the model $\mathcal{A}_\varphi$ need should not be more than L.
- Suppose $(\mathbf{x}_v^{(\ell)})_i$ represents i^{th} component of the feature vector of node v at iteration ℓ.
- $(\mathbf{x}_v^{(\ell)})_\ell$ should evaluate to 1 if and only if the formula φ_ℓ is satisfied at node v.

Graded Modal Logic

Definition

If every bound variable in a FOC_2 formula involves an edge constraints with it we will call them *graded* modal logic. Thus a graded modal formula φ is of form:

$$\varphi(x) : \text{Col}(x) \mid \neg\varphi(x) \mid \varphi(x) \wedge \psi(x) \mid \exists^{\geq n} y(E(x, y) \wedge \varphi(y)). \tag{2}$$

Proposition

Each graded modal logic classifier is captured by a simple homogeneous AC-GNN.

- $\mathcal{A}_\varphi$ satisfies graded modal formula φ.
- Suppose the graded modal logic formula φ involves L sub-formulas $(\varphi_1, \varphi_2, \ldots, \varphi_L)$ such that if φ_k is a sub-formula of φ_l then $k < l$.

- Embedding dimension of each node $\boldsymbol{x}_v$ to be L.
- The number of iterations the model $\mathcal{A}_\varphi$ need should not be more than L.
- Suppose $(\mathbf{x}_v^{(\ell)})_i$ represents i^{th} component of the feature vector of node v at iteration ℓ.
- $(\mathbf{x}_v^{(\ell)})_\ell$ should evaluate to 1 if and only if the formula φ_ℓ is satisfied at node v.

- $(\mathbf{x}_v^{(L)})_L = 1$ for all the nodes v if and only if φ is satisfied at the node.

Model Construction

What we need to define is the matrices $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{b}$.

$$\mathbf{x}_v^{(i)} = f\left(\boldsymbol{A}\mathbf{x}_v^{(i-1)} + \sum_{u \in \mathcal{N}(v)} \boldsymbol{C}\mathbf{x}_u^{(i-1)} + \boldsymbol{b}\right)$$

$$\mathrm{AGG}(\boldsymbol{U}) = \sum_{\boldsymbol{u} \in \boldsymbol{U}} \boldsymbol{u}$$

$$\mathrm{COM}(\boldsymbol{v}, \boldsymbol{u}) = f(\boldsymbol{A}\boldsymbol{v} + \boldsymbol{C}\boldsymbol{u} + \boldsymbol{b})$$

$$f(x) = \min(\max(0, x), 1) \text{ and } \mathbf{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)}$$

Model Construction

What we need to define is the matrices $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{b}$.

$$\mathbf{x}_v^{(i)} = f\left(\boldsymbol{A}\boldsymbol{x}_v^{(i-1)} + \sum_{u \in \mathcal{N}(v)} \boldsymbol{C}\mathbf{x}_u^{(i-1)} + \boldsymbol{b}\right)$$

$$\mathrm{AGG}(\boldsymbol{U}) = \sum_{\boldsymbol{u} \in \boldsymbol{U}} \boldsymbol{u}$$

$$\mathrm{COM}(\boldsymbol{v}, \boldsymbol{u}) = f(\boldsymbol{A}\boldsymbol{v} + \boldsymbol{C}\boldsymbol{u} + \boldsymbol{b})$$

$$f(x) = \min(\max(0, x), 1) \text{ and } \boldsymbol{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)}$$

Case 0. if $\varphi_\ell(v) = \mathrm{Col}(v)$ with Col one of the (base) colors, then $A_{\ell\ell} = 1$,

Case 1. if $\varphi_\ell(v) = \varphi_j(v) \wedge \varphi_k(v)$ then $A_{j\ell} = A_{k\ell} = 1$ and $b_\ell = -1$,

Case 2. if $\varphi_\ell(v) = \neg\varphi_k(v)$ then $A_{k\ell} = -1$ and $b_\ell = 1$,

Case 3. if $\varphi_\ell(v) = \exists^{\geq n} u(E(v, u) \wedge \varphi_k(u))$ then $C_{k\ell} = 1$ and $b_\ell = -n + 1$

Model Construction

What we need to define is the matrices $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{b}$.

$$\mathbf{x}_v^{(i)} = f\left(\boldsymbol{A}\mathbf{x}_v^{(i-1)} + \sum_{u \in \mathcal{N}(v)} \boldsymbol{C}\mathbf{x}_u^{(i-1)} + \boldsymbol{b}\right)$$

$$\mathrm{AGG}(\boldsymbol{U}) = \sum_{\boldsymbol{u} \in \boldsymbol{U}} \boldsymbol{u}$$

$$\mathrm{COM}(\boldsymbol{v}, \boldsymbol{u}) = f(\boldsymbol{A}\boldsymbol{v} + \boldsymbol{C}\boldsymbol{u} + \boldsymbol{b})$$

$$f(x) = \min(\max(0, x), 1) \text{ and } \mathbf{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)}$$

Case 0. if $\varphi_\ell(v) = \mathrm{Col}(v)$ with Col one of the (base) colors, then $A_{\ell\ell} = 1$,

Case 1. if $\varphi_\ell(v) = \varphi_j(v) \wedge \varphi_k(v)$ then $A_{j\ell} = A_{k\ell} = 1$ and $b_\ell = -1$,

Case 2. if $\varphi_\ell(v) = \neg\varphi_k(v)$ then $A_{k\ell} = -1$ and $b_\ell = 1$,

Case 3. if $\varphi_\ell(v) = \exists^{\geq n} u(E(v, u) \wedge \varphi_k(u))$ then $C_{k\ell} = 1$ and $b_\ell = -n + 1$

Lets take Case 3.
Say by Induction hypothesis (on the length of sub-formula) we know that $(\mathbf{x}_u^{(i-1)})_k = 1$ if and only if $v \models \varphi_k$ and 0 otherwise. So

$$(\mathbf{x}_u^{(i)})_\ell = f\left(-n + 1 + \sum_{(u,v) \in G.E} (\mathbf{x}_u^{(i-1)})_k\right)$$

Model Construction

What we need to define is the matrices $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{b}$.

$$\mathbf{x}_v^{(i)} = f\left(\boldsymbol{A}\mathbf{x}_v^{(i-1)} + \sum_{u \in \mathcal{N}(v)} \boldsymbol{C}\mathbf{x}_u^{(i-1)} + \boldsymbol{b}\right)$$

$$\mathrm{AGG}(\boldsymbol{U}) = \sum_{\boldsymbol{u} \in \boldsymbol{U}} \boldsymbol{u}$$

$$\mathrm{COM}(\mathbf{v}, \boldsymbol{u}) = f(\boldsymbol{A}\mathbf{v} + \boldsymbol{C}\boldsymbol{u} + \boldsymbol{b})$$

$$f(x) = \min(\max(0, x), 1) \text{ and } \mathbf{x}_v^{(0)} = \mathbb{1}_{\mathrm{Col}(v)}$$

Case 0. if $\varphi_\ell(v) = \mathrm{Col}(v)$ with Col one of the (base) colors, then $A_{\ell\ell} = 1$,

Case 1. if $\varphi_\ell(v) = \varphi_j(v) \wedge \varphi_k(v)$ then $A_{j\ell} = A_{k\ell} = 1$ and $b_\ell = -1$,

Case 2. if $\varphi_\ell(v) = \neg\varphi_k(v)$ then $A_{k\ell} = -1$ and $b_\ell = 1$,

Case 3. if $\varphi_\ell(v) = \exists^{\geq n} u(E(v, u) \wedge \varphi_k(u))$ then $C_{k\ell} = 1$ and $b_\ell = -n + 1$

Lets take Case 3.
Say by Induction hypothesis (on the length of sub-formula) we know that $(\mathbf{x}_u^{(i-1)})_k = 1$ if and only if $v \models \varphi_k$ and 0 otherwise. So

$$(\mathbf{x}_u^{(i)})_\ell = f\left(-n + 1 + \sum_{(u,v) \in G.E} (\mathbf{x}_u^{(i-1)})_k\right)$$

Barceló et al. [2020] at ICLR 2020. Refer

Converse

Proposition

If AC-GNN captures a logic formula α, then α is a graded modal formula.

Converse

Proposition

If AC-GNN captures a logic formula α, then α is a graded modal formula.

1. Seeks help of Weisfeiler-Lehman (WL) test – a heuristic when applied to two graphs separately it returns some partitions of vertices for each of the graphs. If they do not agree on each other then we can say the graphs are non-isomorphic.
2. WL-Test and AC-GNN.
3. GML and WL-Test.

Converse (ii)

Proof. We would take help of the following propositions:

1. If the WL test assigns the same color to two nodes in a graph, then every AC-GNN classifies either both nodes as true or both nodes as `false`. [Xu et al., 2018]
2. Let α be a unary FO formula. If α is not equivalent to a graded modal logic formula then there exist two graphs G, G' and two nodes v in G and u' in G' such that $\mathrm{Unrv}_G^L(v) \simeq$ isomorphic $\mathrm{Unrv}_{G'}^L(u')$ for every $L \in \mathbb{N}$ and such that $u \models \alpha$ in G but $u' \not\models \alpha$ in G'. [Otto, 2019].

Let G be a graph (simple, undirected and node-colored), v be a node in G, and $L \in \mathbb{N}$. The unravelling of v in G at depth L, denoted by $\mathrm{Unrv}_G^L(v)$, is the (simple undirected nodecolored) graph that is the tree having:

- a node $(v, u_1, \ldots, u_i)$ for each path $(v, u_1, \ldots, u_i)$ in G with $i \leq L$,
- an edge between $(v, u_1, \ldots, u_{i-1})$ and $(v, u_1, \ldots, u_i)$ when $\{u_{i-1}, u_i\}$ is an edge in G (assuming that u_0 is v), and
- each node $(v, u_1, \ldots, u_i)$ colored the same as u_i in G.

Then we can observe that

3. Let G and G' be two graphs, and v and v' be two nodes in G and G', respectively. Then for every $L \in \mathbb{N}$, the WL test assigns the same color to v and v' at round L if and only if there is an isomorphism between $\mathrm{Unrv}_G^L(v)$ and $\mathrm{Unrv}_{G'}^L(v')$ sending v to v'.

Now from observation 1 and 3, we can conclude the follow:

4. Let G and G' be two graphs with nodes v in G and v' in G' such that $\mathrm{Unrv}_G^L(v)$ is isomorphic to $\mathrm{Unrv}_{G'}^L(v')$ for every $L \in \mathbb{N}$. Then for any AC-GNN $\mathcal{A}$, we have $\mathcal{A}(G, u) = \mathcal{A}(G', u')$.

We will now prove the contrapositive of the statement to be proved – If a logical classifier α is not equivalent to any graded modal logic formula, then there is no AC-GNN that captures α.

Let α be a logical classifier that is not equivalent to any graded modal logic formula. To contradict, let us assume there exists an AC-GNN $\mathcal{A}_\alpha$ capturing α. By observation 2, there exist two graphs G, G' and two nodes v in G and u' in G' such that i) $\mathrm{Unrv}_G^L(v) \simeq \mathrm{Unrv}_{G'}^L(u')$ for every $L \in \mathbb{N}$ and ii) such that $u \models \alpha$ in G but $u' \not\models \alpha$ in G'. But as i) holds by observation 4, we can say $\mathcal{A}(G, u) = \mathcal{A}(G', u')$ – violating to ii) and hence the contradiction that $\mathcal{A}_\alpha$ can capture α. □

Otto [2019], Xu et al. [2018]

References

Pablo Barceló, Egor V. Kostylev, Mikael Monet, Jorge Pérez, Juan Reutter, and Juan Pablo Silva. The logical expressiveness of graph neural networks. In *International Conference on Learning Representations*, 2020. URL *https://openreview.net/forum?id=r1lZ7AEKvB*.

Martin Otto. Graded modal logic and counting bisimulation. *ArXiv*, abs/1910.00039, 2019. URL *https://api.semanticscholar.org/CorpusID:203610381*.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *ArXiv*, abs/1810.00826, 2018. URL *https://api.semanticscholar.org/CorpusID:52895589*.

Thank You