A Appendix

A.1 Preliminaries for Proofs

In this section, we give some preliminaries which will be used to prove the theorems, proposition and lemmas shown in our main body. In what follows, we fix a unary predicate set P_1 and a binary predicate set P_2.

Definition 16. *A R^2-GNN is defined to be 0/1-GNN if the recursive formula used to compute vectors $\mathbf{x}_v^{(i)}$ for each node v of a multi-edge graph $G = \{V, \mathcal{E}, P_1, P_2\}$ on each layer i is in the following form*

$$\boldsymbol{x}_v^{(i)} = f\left(C^{(i)}\left(\mathbf{x}_v^{(i-1)} + \sum_{r\in P_2}\sum_{u\in V} A_r^{(i)}\mathbf{x}_u^{(i-1)} + R^{(i)}\left(\sum_{u\in V}\boldsymbol{x}_u^{(i-1)}\right) + b^{(i)}\right)\right) \tag{5}$$

where $C^{(i)}$,$A_j^{(i)}$,$R^{(i)}$ are all integer matrices of size $d_i \times d_{i-1}$, $b^{(i)}$ is bias column vector with size $d_i \times 1$, where d_{i-1} and d_i are input/output dimensions, and f is defined as $max(0,min(x,1))$.

Furthermore, we restrict the final output dimension be $d_L = 1$. Since all matrices have integer elements, initial vectors are set to integers by initialisation function $I(\cdot)$, and $max(0,min(x,1))$ will map all integers to $0/1$, it's easy to see that the output of this kind of model is always $0/1$, which can be directly used as the classification result. We call such model 0/1-GNN. A model instance can be represented by $\{C^{(i)},(A_j^{(i)})_{j=1}^K,R^{(i)},b^{(i)}\}_{i=1}^L$

Lemma 17. *Regard 0/1-GNN as node classifier, then the set of node classifiers represented by 0/1-GNN is closed under $\wedge$, $\vee$,$\neg$.*

Proof. Given two 0/1-GNN $\mathcal{A}_1$,$\mathcal{A}_2$, it suffices to show that we can construct $\neg\mathcal{A}_1$ and $\mathcal{A}_1 \wedge \mathcal{A}$ in 0/1-GNN framework. Notice that $\vee$ can be reducted to $\wedge$,$\neg$ by De Morgan's law, e.g., $a \vee b = \neg(\neg a \wedge \neg b)$.

1. Construct $\neg\mathcal{A}_1$. Append a new layer to $\mathcal{A}_1$ with dimension $d_{L+1} = 1$. For matrices and bias $C^{(L+1)}$,$(A_j^{(L+1)})_{j=1}^K$,$R^{(L+1)}$,$b^{(L+1)}$ in layer $L + 1$, set $C_{1,1}^{L+1} = -1$ and $b_1^{L+1} = 1$ and other parameters 0. Then it follows $\mathbf{x}_v^{(L+1)} = max(0,min(-\mathbf{x}_v^{(L)} + 1,1))$. Since $\mathbf{x}_v^{(L)}$ is the 0/1 classification result outputted by $\mathcal{A}_1$. It's easy to see that the above equation is exactly $\mathbf{x}_v^{(L+1)} = \neg\mathbf{x}_v^{(L)}$

2. Construct $\mathcal{A}_1 \wedge \mathcal{A}_2$. Without loss of generality, we can assume two models have same layer number L and same feature dimension d_l in each layer $l \in \{1......L\}$. Then, we can construct a new 0/1-GNN $\mathcal{A}$. $\mathcal{A}$ has $L + 1$ layers. For each of the first L layers, say l-th layer, it has feature dimension $2d_l$. Let $\{C_1^{(l)},(A_{j,1}^{(l)})_{j=1}^K,R_1^{(l)},b_1^{(l)}\}$,$\{C_2^{(l)},(A_{j,2}^{(l)})_{j=1}^K,R_2^{(l)},b_2^{(l)}\}$ be parameters in layer l of $\mathcal{A}_1$,$\mathcal{A}_2$ respectively. Parameters for layer l of $\mathcal{A}$ are defined below

$$\mathbf{C}^{(l)} := \begin{bmatrix}\mathbf{C}_1^{(l)} & \\ & \mathbf{C}_2^{(l)}\end{bmatrix} \mathbf{A}_j^{(l)} := \begin{bmatrix}\mathbf{A}_{j,1}^{(l)} & \\ & \mathbf{A}_{j,2}^{(l)}\end{bmatrix} \mathbf{R}^{(l)} := \begin{bmatrix}\mathbf{R}_1^{(l)} & \\ & \mathbf{R}_2^{(l)}\end{bmatrix} \mathbf{b}^{(l)} := \begin{bmatrix}\mathbf{b}_1^{(l)} \\ \mathbf{b}_2^{(l)}\end{bmatrix} \tag{6}$$

Initialization function of $\mathcal{A}$ is concatenation of initial feature of $\mathcal{A}_1$,$\mathcal{A}_2$. Then it's easy to see that the feature $\mathbf{x}_v^L$ after running first L layers of $\mathcal{A}$ is a two dimension vector, and the two dimensions contains two values representing the classification results outputted by $\mathcal{A}_1$,$\mathcal{A}_2$ respectively.

For the last layer $L+1$, it has only one output dimension. We just set $\mathbf{C}_{1,1}^{L+1} = \mathbf{C}_{1,2}^{L+1} = 1$,$\mathbf{b}_1^{L+1} = -1$ and all other parameters 0. Then it's equivalent to $\mathbf{x}_v^{(L+1)} = max(0,min(\mathbf{x}_{v,1}^{(L)} + \mathbf{x}_{v,2}^{(L)} - 1,1))$ where $\mathbf{x}_{v,1}^{(L)}$,$\mathbf{x}_{v,2}^{(L)}$ are output of $\mathcal{A}_1$,$\mathcal{A}_2$ respectively. It's easy to see that the above equation is equivalent to $\mathbf{x}_v^{(L+1)} = \mathbf{x}_{v,1}^{(L)} \wedge \mathbf{x}_{v,2}^{(L)}$ so the $\mathcal{A}$ constructed in this way is exactly $\mathcal{A}_1 \wedge \mathcal{A}_2$ $\square$

Definition 18. *A $\mathcal{FOC}_2$ formula is defined inductively according to the following grammar:*

$$A(x), r(x,y), \varphi_1 \wedge \varphi_2, \varphi_1 \vee \varphi_2, \neg\varphi_1, \exists^{\geq n} y(\varphi_1(x,y)) \text{ where } A \in P_1 \text{ and } r \in P_2 \tag{7}$$

Definition 19. *For any subset $S \subseteq P_2$, let $\varphi_S(x,y)$ denote the $\mathcal{FOC}_2$ formula $(\bigwedge_{r\in S} r(x,y)) \wedge (\bigwedge_{r\in P_2\setminus S} \neg r(x,y))$. Note that $\varphi_S(x,y)$ means there is a relation r between x and y if and only if*

$r \in S$, so $\varphi_S(x,y)$ can be seen as a formula to restrict specific relation distribution between two nodes. $\mathcal{RSFOC}_2$ is inductively defined according to the following grammar:

$$A(x), \varphi_1 \wedge \varphi_2,\ \varphi_1 \vee \varphi_2,\ \neg\varphi_1, \exists^{\geq n} y\Big(\varphi_S(x,y) \wedge \varphi_1(y)\Big) \text{ where } A \in P_1 \text{ and } S \subseteq P_2 \tag{8}$$

Next, we prove that $\mathcal{FOC}_2$ and $\mathcal{RSFOC}_2$ have the same expressiveness, namely, each $\mathcal{FOC}_2$ node classifier can be rewritten in the form $\mathcal{RSFOC}_2$.

Lemma 20. $\mathcal{FOC}_2 = \mathcal{RSFOC}_2$.

Proof. Comparing the definitions of $\mathcal{RSFOC}_2$ and $\mathcal{FOC}_2$, it is obvious that $\mathcal{RSFOC}_2 \subseteq \mathcal{FOC}_2$ trivially holds, so we only need to prove the other direction, namely, $\mathcal{FOC}_2 \subseteq \mathcal{RSFOC}_2$. In particular, a Boolean logical classifier only contains one free variable, we only need to prove that for any one-free-variable $\mathcal{FOC}_2$ formula $\varphi(x)$, we can construct an equivalent $\mathcal{RSFOC}_2$ formula $\psi(x)$.

We prove Lemma 20 by induction over k, where k is the quantifier depth of $\varphi(x)$.

In the base case where $k = 0$, $\varphi(x)$ is just the result of applying conjunction, disjunction or negation to a bunch of unary predicates $A(x)$, where $A \in P_1$. Given that the grammar of generating $\varphi(x)$ is the same in $\mathcal{RSFOC}_2$ and $\mathcal{FOC}_2$ when $k = 0$, so the lemma holds for $k = 0$.

For the indutive step, we assume that Lemma 20 holds for all $\mathcal{RSFOC}_2$ formula with quantifier depth no more than m, we next need to consider the case when $k = m + 1$.

We can decompose $\varphi(x)$ to be boolean combination of a bunch of $\mathcal{FOC}_2$ formulas $\varphi_1(x), \ldots, \varphi_N(x)$, each of which is in the form $\varphi_i(x) := A(x)$ where $A \in P_1$ or $\varphi_i(x) := \exists^{\geq n} y(\varphi'(x,y))$. See the following example for reference.

Example 21. *Assume $\varphi(x) := \big(A_1(x) \wedge \exists y(r_1(x,y))\big) \vee \big(\exists y\big(A_2(y) \wedge r_2(x,y)\big) \wedge \exists y(r_3(x,y))\big)$. It can be decomposed into boolean combination of four subformulas shown as follows:*

- $\varphi_1(x) = A_1(x)$
- $\varphi_2(x) = \exists y(r_1(x,y))$
- $\varphi_3(x) = \exists y\big(A_2(y) \wedge r_2(x,y)\big)$
- $\varphi_4(x) = \exists y(r_3(x,y))$

We can see that grammars of $\mathcal{FOC}_2$ and $\mathcal{RSFOC}_2$ have a common part: $A(x), \varphi_1 \wedge \varphi_2, \varphi_1 \vee \varphi_2, \neg\varphi_1$, so we can only focus on those subformulas $\varphi_i(x)$ in the form of $\exists^{\geq n} y \varphi'(x,y)$. In other words, if we can rewrite these $\mathcal{FOC}_2$ subformulas into another form satisfying the grammar of $\mathcal{RSFOC}_2$, we can naturally construct the desired $\mathcal{RSFOC}_2$ formula $\psi(x)$ equivalent to $\mathcal{FOC}_2$ formula $\varphi(x)$.

Without loss of generality, in what follows, we consider the construction for $\varphi(x) = \exists^{\geq n} y(\varphi'(x,y))$. Note that $\varphi(x)$ has quantifier depth no more than $m + 1$, and $\varphi'(x,y)$ has quantifier depth no more than m.

We can decompose $\varphi'(x,y)$ into three sets of subformulas $\{\varphi_i^x(x)\}_{i=1}^{N_x}, \{\varphi_i^y(y)\}_{i=1}^{N_y}, \{r_i(x,y)\}_{i=1}^{|P_2|}$, where N_x and N_y are two natural numbers, φ_i^x, φ_i^y are its maximal subformulas whose free variable is assigned to x and y, respectively. $\varphi'(x)$ is the combination of these sets of subformulas using $\wedge, \vee, \neg$.

Example 22. *Assume that we have a $\mathcal{FOC}_2$ formula in the form of $\varphi'(x,y) = \Big(r_1(x,y) \wedge \exists x(r_2(x,y))\Big) \vee \Big(\exists y\big(\exists x(r_3(x,y)) \vee \exists y(r_1(x,y))\big) \wedge \exists y\big(A_2(y) \wedge r_2(x,y)\big)\Big)$*

It can be decomposed into the following subformulas:

- $\varphi_1^x(x) := \exists y\big(\exists x(r_3(x,y)) \vee \exists y(r_1(x,y))\big)$;
- $\varphi_2^x(x) := \exists y\big(A_2(y) \wedge r_2(x,y)\big)$;
- $\varphi_1^y(y) := \exists x(r_2(x,y))$;

- $r_1(x,y)$

Assume that $N := \{1,\ldots,N_x\}$, we construct a $\mathcal{RSFOC}_2$ formula $\varphi^x_T(x) := (\bigwedge_{i\in T}\varphi^x_i(x)) \wedge (\bigwedge_{i\in N\setminus T}\neg\varphi^x_i(x))$, where $T\subseteq N$. It is called the *x-specification* formula, which means $\varphi^x_T(x)$ is *true* iff the following condition holds: for all $i\in T$, $\varphi^x_i(x)$ is *true* and for all $i\in N\setminus T$, $\varphi^x_i(x)$ is *false*.

By decomposing $\varphi'(x,y)$ into three subformula sets, we know Boolean value of $\varphi'(x,y)$ can be decided by Boolean values of these formulas $\{\varphi^x_i(x)\}_{i=1}^{N_x}$,$\{\varphi^y_i(y)\}_{i=1}^{N_y}$,$\{r_i(x,y)\}_{i=1}^{|P_2|}$. Now for any two specific subsets $S\subseteq P_2, T\subset N$, we assume $\varphi_S(x,y)$ and $\varphi^x_T(x)$ are all *true* (Recall the definition of $\varphi_S(x,y)$ in Definition 19). Then Boolean values for formulas in $\{\varphi^x_i(x)\}_{i=1}^{N_x}$,$\{r_i(x,y)\}_{i=1}^{|P_2|}$ are determined and Boolean value of $\varphi'(x,y)$ depends only on Boolean values of $\{\varphi^y_i(y)\}_{i=1}^{N_y}$. Therefore, we can write a new $\mathcal{FOC}_2$ formula $\varphi^y_{S,T}(y)$ which is a boolean combination of $\{\varphi^y_i(y)\}_{i=1}^{N_y}$. This formula should satisfy the following condition: For any graph G and two nodes a,b on it, the following holds,

$$\varphi_S(a,b)\wedge\varphi^x_T(a)\Rightarrow\Big(\varphi'(a,b)\Leftrightarrow\varphi^y_{S,T}(b)\Big) \tag{9}$$

By our inductive assumption, $\varphi'(x,y)$ has a quantifier depth which is no more than m, so $\{\varphi^x_i(y)\}_{i=1}^{N_x}$,$\{\varphi^y_i(y)\}_{i=1}^{N_y}$ also have quantifier depths no more than m. Therefore, each of them has $\mathcal{RSFOC}_2$ correspondence. Furthermore, since $\wedge$, $\vee$,$\neg$ are allowed operation in $\mathcal{RSFOC}_2$, $\varphi^x_T(x)$ and $\varphi^y_{S,T}(y)$ can also be rewritten as $\mathcal{RSFOC}_2$ formulas.

Given that $\varphi_S(x,y)$ and $\varphi^x_T(y)$ specify the boolean values for all $\{\varphi^x_i(y)\}_{i=1}^{N_x}$,$\{\varphi^r_i(x,y)\}_{i=1}^{|P_2|}$ formulas, so we can enumerate all possibilities over $S\subseteq P_2$ and $T\subseteq N$. Obviously for any graph G and a node pair (a,b), there exists an unique (S,T) pair such that $\varphi_S(a,b)\wedge\varphi^x_T(a)$ holds.

Hence, combining Equation (9), $\varphi'(x,y)$ is true only when there exists a (S,T) pair such that $\varphi_S(x,y)\wedge\varphi^x_T(x)\wedge\varphi^y_{S,T}(y)$ is *true*. Formally, we can rewrite $\varphi'(x,y)$ as following form:

$$\varphi'(x,y)\equiv\bigvee_{S\subseteq P_2,T\subseteq N}\Big(\varphi_S(x,y)\wedge\varphi^x_T(x)\wedge\varphi^y_{S,T}(y)\Big) \tag{10}$$

In order to simplify the formula above, let $\phi_T(x)$ denote the following formula:

$$\phi_T(x,y) \coloneqq \bigvee_{S\subseteq P_2}\Big(\varphi_S(x,y)\wedge\varphi^y_{S,T}(y)\Big) \tag{11}$$

Then we can simplify Equation (10) to the following form:

$$\varphi'(x,y)\equiv\bigvee_{T\subseteq N}\Big(\varphi^x_T(x)\wedge\phi_T(x,y)\Big) \tag{12}$$

Recall that $\varphi(x)=\exists^{\geq n}y(\varphi'(x,y))$, so it can be rewritten as:

$$\varphi(x)\equiv\exists^{\geq n}y\bigg(\bigvee_{T\subseteq N}\big(\varphi^x_T(x)\wedge\phi_T(x,y)\big)\bigg) \tag{13}$$

Since for any graph G and its node a, there exists exactly one T such that $\varphi^x_T(a)$ is *true*. Therefore, Equation (13) can be rewritten as the following formula:

$$\varphi(x)\equiv\bigvee_{T\subseteq N}\bigg(\varphi^x_T(x)\wedge\exists^{\geq n}y(\phi_T(x,y))\bigg) \tag{14}$$

Let $\widehat{\varphi}_T(x) \coloneqq \exists^{\geq n}y(\phi_T(x,y))$. Since $\wedge$,$\vee$ are both allowed in $\mathcal{RSFOC}_2$. If we want to rewrite $\varphi(x)$ in the $\mathcal{RSFOC}_2$ form, it suffices to rewrite $\widehat{\varphi}_T(x)$ as a $\mathcal{RSFOC}_2$ formula, which is shown as follows,

$$\widehat{\varphi}_T(x) \coloneqq \exists^{\geq n}y(\phi_T(x,y))=\exists^{\geq n}y\bigg(\bigvee_{S\subseteq P_2}\Big(\varphi_S(x,y)\wedge\varphi^y_{S,T}(y)\Big)\bigg) \tag{15}$$

Similar to the previous argument, since for any graph G and of of its node pairs (a,b), the *relation-specification* formula $\varphi_S(x,y)$ restricts exactly which types of relations exists between (a,b), there is exactly one subset $S \subseteq P_2$ such that $\varphi_S(a,b)$ holds.

Therefore, for all $S \subseteq P_2$, we can define n_S as the number of nodes y such that $\varphi_S(x,y) \wedge \varphi^y_{S,T}(y)$ holds. Since for two different subsets $S_1,S_2 \subseteq P_2$ and a fixed y, $\varphi_{S_1}(x,y)$ and $\varphi_{S_2}(x,y)$ can't hold simultaneously, the number of nodes y that satisfies $\varphi_S(x,y) \wedge \varphi^y_{S,T}(y)$ is exactly the sum $\sum_{S\subseteq P_2} n_S$. Therefore, in order to express Equation (15), which means there exists at least n nodes y such that $\bigvee_{S\subseteq P_2}\big(\varphi_S(x,y) \wedge \varphi^y_{S,T}(y)\big)$ holds, it suffices to enumerate all possible values for $\{n_S | S \subseteq P_2\}$ that satisfies $(\sum_{S\subseteq P_2} n_S) = n, n_S \in \mathbb{N}$. Formally, we can rewrite $\widehat{\varphi}_T(x)$ as follows:

$$\widehat{\varphi}_T(x) \equiv \bigvee_{(\sum_{S\subseteq P_2} n_S)=n} \Big(\bigwedge_{S\subseteq P_2} \exists^{\geq n_S} y(\varphi_S(x,y) \wedge \varphi^y_{S,T}(y)) \Big) \tag{16}$$

Note that $\exists^{\geq n_S} y(\varphi_S(x,y) \wedge \varphi^y_{S,T}(y))$ satisfies the grammar of $\mathcal{RSFOC}_2$, so $\widehat{\varphi}_T(x)$ can be rewritten as $\mathcal{RSFOC}_2$. Then, since $\varphi^x_T(x)$ can also be rewritten as $\mathcal{RSFOC}_2$ by induction, combining Equation (14) and Equation (15), $\varphi(x)$ is in $\mathcal{RSFOC}_2$. We finish the proof. $\square$

A.2 Proof of Proposition 2

Proposition 2. *$\mathcal{FOC}_2 \not\subseteq$ R^2-GNNs and R^2-GNNs $\not\subseteq \mathcal{FOC}_2$ on some* universal *graph class $\mathcal{G}_u$.*

Proof. First, we prove $\mathcal{FOC}_2 \not\subseteq$ R^2-GNNs.

Consider the two graphs G_1,G_2 in Figure 1. $(G_1,a),(G_2,a)$ can be distinguished by the $\mathcal{FOC}_2$ formula $\varphi(x) := \exists^{\geq 1} y(p_1(x,y) \wedge p_2(x,y))$. However, we will prove that any R^2-GNN can't distinguish any node in G_1 from any node in G_2.

Let's prove it by induction over the layer number L of R^2-GNN. That's to say, we want to show that for any $L \geq 0$, R^2-GNN with no more than L layers can't distinguish any node of G_1 from that of G_2.

For the base case where $L = 0$, since each node feature vector is initialized by the unary predicate information, so the result trivially holds.

Assume any R^2-GNN with no more than $L = m$ layers can't distinguish nodes of G_1 from nodes of G_2. Then we want to prove the result for $L = m + 1$.

For any R^2-GNN model $\mathcal{A}$ with $m + 1$ layers, let $\mathcal{A}'$ denote its first m layers, we know outputs of $\mathcal{A}'$ on any node from G_1 or G_2 are the same, suppose the common output feature is $\mathbf{x}^{(m)}$.

Recall the updating rule of R^2-GNN in Equation (2).We know the output of $\mathcal{A}$ on any node v in G_1 or G_2 is defined as follows,

$$\mathbf{x}_v^{(m+1)} = C^{(m+1)}\bigg(\mathbf{x}_v^{(m)}, \Big(A_1^{(m+1)}(\{\!\{\mathbf{x}_{u_1(v)}^{(m)}\}\!\})\Big), A_2^{(m+1)}(\{\!\{\mathbf{x}_{u_2(v)}^{(m)}\}\!\})\Big), R^{(m+1)}(\{\!\{\mathbf{x}_a^{(m)}, \mathbf{x}_b^{(m)}, \mathbf{x}_c^{(m)}, \mathbf{x}_d^{(m)}\}\!\})\bigg) \tag{17}$$

Here $C^{(m+1)}, A_1^{(m+1)}, A_2^{(m+1)}, R^{(m+1)}$ are parameters in the layer $m + 1$ of $\mathcal{A}$, $u_1(v),u_2(v)$ is the only r_1,r_2-type neighbor of v, and a,b,c,d are nodes from the corresponding graph G_1 or G_2. From Figure 1 we can see they are well defined.

By induction, since any node pairs from G_1 and G_2 can't be distinguished by $\mathcal{A}'$, we have $\mathbf{x}_v^{(m)}, \mathbf{x}_{u_1(v)}^{(m)}, \mathbf{x}_{u_2(v)}^{(m)}, \mathbf{x}_a^{(m)}, \mathbf{x}_b^{(m)}, \mathbf{x}_c^{(m)}, \mathbf{x}_d^{(m)}$ are all the same feature $\mathbf{x}^{(m)}$. Therefore, Equation (17) have the same expression for all nodes v from G_1 and G_2, which implies any $\mathcal{A}$ with $m + 1$ layers can't distinguish nodes from G_1 and G_2.

Next, we then prove R^2-GNNs $\not\subseteq \mathcal{FOC}_2$.

Assume we want to construct a classifier c which classifies a node into true iff *the node has a larger number of r_1-type neighbors than that of r_2-type neighbors.*

First, we prove that we can construct an 0/1-GNN $\mathcal{A}$ to capture c. It only has one layer with parameters $C^{(1)}, A_1^{(1)}, A_2^{(1)}, R^{(1)}$, and feature dimension $d_0 = d_1 = 1$. We assume that each node has

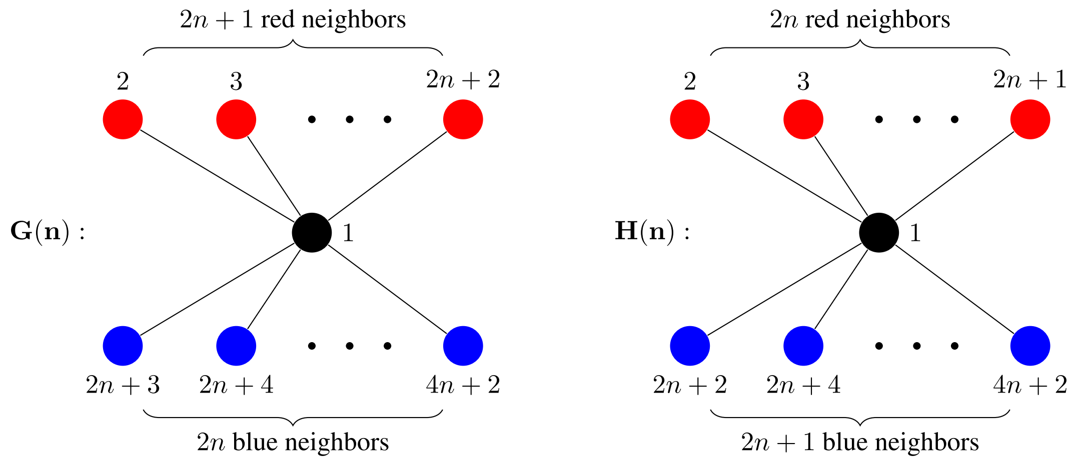

Figure 5: $G(n)$ and $H(n)$.

the same initial feature vector, i.e., $\mathbf{1}$. We set $A^{(1)}_{1,(1,1)} = 1, A^{(1)}_{2,(1,1)} = -1$, where $A^{(1)}_{1,(1,1)}$ denotes the only element in $A^{(1)}_1$ placed in the first row and first column (similar for $A^{(1)}_{2,(1,1)}$) and all other parameters 0. It's easy to see that $\mathcal{A}$ is equivalent to our desired classifier c on any graph since we have $\mathbf{x}^{(1)}_v = max(0, min(1, \sum_{u \in \mathcal{N}_{G,1}(v)} 1 - \sum_{u \in \mathcal{N}_{G,2}(v)} 1))$.

Next, we show $\mathcal{FOC}_2$ can't capture c on $\mathcal{G}_s$. In order to show that, for any natural number n, we can construct two single-edge graphs $G(n), H(n)$ as follows:

$$
\begin{aligned}
V(G(n)) &= V(H(n)) = \{1,2......4n+2\} \\
E(G(n)) &= \{r_1(1,i) | \forall i \in [2, 2n+2]\} \cup \{r_2(1,i) | i \in [2n+3, 4n+2]\} \\
E(H(n)) &= \{r_1(1,i) | \forall i \in [2, 2n+1]\} \cup \{r_2(1,i) | i \in [2n+2, 4n+2]\}
\end{aligned}
$$

We prove the result by contradiction. Assume there is a $\mathcal{FOC}_2$ classifier φ that captures the classifier c, then it has to classify $(G(n), 1)$ as *true* and $(H(n), 1)$ as *false* for all natural number n. However, in the following we will show that it's impossible, which proves the non-existence of such φ.

Suppose threshold numbers used on counting quantifiers of φ don't exceed m, then we only need to prove that φ can't distinguish $(G(m), 1), (H(m), 1)$, which contradicts our assumption.

For simplicity, we use G, H to denote $G(m), H(m)$. In order to prove the above argument. First, we define a *node-classification* function $CLS(\cdot)$ as follows. It has G or H as subscript and a node of G or H as input.

1. $CLS_G(1) = CLS_H(1) = 1$. It means the function returns 1 when the input is the *center* of G or H.
2. $CLS_G(v_1) = CLS_H(v_2) = 2, \forall v_1 \in [2, 2m+2], \forall v_2 \in [2, 2m+1]$, which means the function returns 2 when the input is a r_1-neighbor of *center*.
3. $CLS_G(v_1) = CLS_H(v_2) = 3, \forall v_1 \in [2m+3, 4m+2], \forall v_2 \in [2m+2, 4m+2]$, which means the function returns 3 when the input is a r_2-neighbor of *center*.

Claim 1: Given any $u_1, v_1 \in V(G), u_2, v_2 \in V(H)$, if $(CLS_G(u_1), CLS_G(v_1)) = (CLS_H(u_2), CLS_H(v_2))$, then any $\mathcal{FOC}_2$ formula with threshold numbers no larger than m can't distinguish (u_1, v_1) and (u_2, v_2).

This claim is enough for our result. We will prove that for any constant d and any $\mathcal{FOC}_2$ formula ϕ with threshold numbers no larger than m and quantifier depth d, ϕ can't distinguish (u_1, v_1) and (u_2, v_2) given that $(CLS_G(u_1), CLS_G(v_1)) = (CLS_H(u_2), CLS_H(v_2))$

The result trivially holds for the base case where $d = 0$. Now let's assume the result holds for $d \leq k$, we can now prove the inductive case when $d = k + 1$.

Since $\wedge,\vee,\neg,r(x,y)$ trivially follows, we can only consider the case when $\phi(x,y)$ is in the form $\exists^{\geq N}y\phi'(x,y),N \leq m$ or $\exists^{\geq N}x\phi'(x,y),N \leq m$, where $\phi'(x,y)$ is a $\mathcal{FOC}_2$ formula with threshold numbers no more than m and quantifier depth no more than k. Since these two forms are symmetrical, without loss of generality, we only consider the case $\exists^{\geq N}y\phi'(x,y),N \leq m$.

Let N_1 denote the number of nodes $v_1' \in V(G)$ such that $(G,u_1,v_1') \models \phi'$ and N_2 denote the number of nodes $v_2' \in V(H)$ such that $(H,u_2.v_2') \models \phi'$. Let's compare values of N_1 and N_2. First, By induction, since we have $CLS_G(u_1) = CLS_H(u_2)$ from precondition, so for any $v_1' \in V(G), v_2' \in V(H)$, which satisfies $CLS_G(v_1') = CLS_H(v_2')$, $\phi'(x,y)$ can't distinguish (u_1,v_1') and (u_2,v_2'). Second, isomorphism tells us ϕ' can't distinguish node pairs from the same graph if they share the same CLS values. Combining these two facts, there has to be a subset $S \subseteq \{1,2,3\}$, such that $N_1 = \sum_{a\in S} N_G(a)$ and $N_2 = \sum_{a\in S} N_H(a)$, where $N_G(a)$ denotes the number of nodes u on G such that $CLS_G(u) = a$, ($N_H(a)$ is defined similarly).

It's easy to see that $N_G(1) = N_H(1) = 1$, and $N_G(a),N_H(a) > m$ for $a \in \{2,3\}$. Therefore, at least one of $N_1 = N_2$ and $m < min\{N_1,N_2\}$ holds. In neither case $\exists^{\geq N}y\phi'(x,y),N \leq m$ can distuigush (u_1,v_1) and (u_2,v_2). □

A.3 Proof of Theorem 3

Theorem 3. *$FOC_2 \subseteq$ R^2-GNNs on any single-edge graph class $\mathcal{G}_s$.*

Proof. By Lemma 20, $\mathcal{FOC}_2 = \mathcal{RSFOC}_2$, so it suffices to show $\mathcal{RSFOC}_2 \subseteq$ 0/1-GNN. By Lemma 17, 0/1-GNN is closed under $\wedge$, $\vee$,$\neg$, so we can only focus on formulas in $\mathcal{RSFOC}_2$ of form $\varphi(x) = \exists^{\geq n}y(\varphi_S(x,y) \wedge \varphi'(y)),S \subseteq P_2$. If we can construct an equivalent 0/1-GNN $\mathcal{A}$ for all formulas of above form, then we can capture all formulas in $\mathcal{RSFOC}_2$ since other generating rules $\wedge$, $\vee$,$\neg$ is closed under 0/1-GNN. In particular, for the setting of *single-edge* graph class, φ is meaningful only when $|S| \leq 1$. That's because $|S| > 2$ implies that φ is just the trivial $\perp$ in any *single-edge* graph class $\mathcal{G}_s$.

Do induction over quantifier depth k of $\varphi(x)$. In the base case where $k = 0$, the result trivially holds since in this situation, the only possible formulas that needs to consider are unary predicates $A(x)$, where $A \in P_1$, which can be captured by the initial one-hot feature. Next, assume our result holds for all formulas with quantifier depth k no more than m, it suffices to prove the result when quantifier depth of $\varphi(x) = \exists^{\geq n}y(\varphi_S(x,y) \wedge \varphi'(y))$ is $m + 1$. It follows that quantifier depth of $\varphi'(y)$ is no more than m.

By induction, there is a 0/1-GNN model $\mathcal{A}'$ such that $\mathcal{A}' = \varphi'$ on single-edge graph class. To construct $\mathcal{A}$, we only need to append another layer on $\mathcal{A}'$. This layer $L + 1$ has dimension 1, whose parameters $C^{(L+1)},(A_j^{(L+1)})_{j=1}^K,R^{(L+1)},b^{(L+1)}$ are set as follows:

1. When $|S| = 1$: Suppose $S = \{j\}$, set $A_{j,(1,1)}^{L+1} = 1,b^{L+1} = 1 - n$, where $A_{j,(1,1)}^{L+1}$ denotes the element on the first row and first column of matrix $A_j^{(L+1)}$. Other parameters in this layer are 0. This construction represents $\mathbf{x}_v^{(L+1)} = max(0,min((\sum_{u\in\mathcal{N}_{G,j}(v)} \mathbf{x}_u^{(L)}) - (n-1),1))$. Since $\mathbf{x}_u^{(L)}$ is classification result outputted by $\mathcal{A}'$ which is equivalent to φ', $\sum_{u\in\mathcal{N}_{G,j}(v)} \mathbf{x}_u^{(L)}$ counts the number of j-type neighbor u of v that satisfies $\varphi'(u)$. Therefore $\mathbf{x}_v^{(L+1)} = 1$ if and only if there exists at least n j-type neighbors satisfying the condition φ', which is exactly what $\varphi(x)$ means.

2. When $|S| = 0$: Let $K = |P_2|$, for all $j \in [K]$, set $A_{j,(1,1)}^{L+1} = -1$, $R_{1,1}^{(L+1)} = 1,b^{L+1} = 1 - n$ and all other parameters 0. This construction represents $\mathbf{x}_v^{(L+1)} = max(0,min((\sum_{u\in V(G)} \mathbf{x}_u^{(L)}) - (\sum_{j=1}^K \sum_{u\in\mathcal{N}_{G,j}(v)} \mathbf{x}_u^{(L)}) - (n-1),1))$. Since we only consider single-edge graph, $(\sum_{u\in V(G)} \mathbf{x}_u^{(L)}) - (\sum_{j=1}^K \sum_{u\in\mathcal{N}_{G,j}(v)} \mathbf{x}_u^{(L)})$ exactly counts the number of nodes u that satisfies $\varphi'(y)$ and doesn't have any relation with v. It's easy to see that $\mathbf{x}_v^{(L+1)} = 1$ iff there exists at least n such nodes u, which is exactly what $\varphi(x)$ means.

Hence, we finish the proof for Theorem 3 – for each $\mathcal{FOC}_2$ formula over the single-edge graph class, we can construct an R^2-GNN to capture it.

$\square$

A.4 Proof of Theorem 4

Theorem 4. *R^2-GNNs $\subseteq$ FOC_2 on any bounded graph class $\mathcal{G}_b$.*

If we want to prove R^2-GNN $\subseteq \mathcal{FOC}_2$, it suffices to show that for any R^2-GNN $\mathcal{A}$, there exists an equivalent $\mathcal{FOC}_2$ formula φ on any bounded graph class $\mathcal{G}_b$. It implies that for two graphs G_1,G_2 and their nodes a,b, if they are classified differently by $\mathcal{A}$, there exists some $\mathcal{FOC}_2$ formula φ that can distinguish them. Conversely, if a,b can't be distinguished by any $\mathcal{FOC}_2$ formula, then they can't be distinguished by any R^2-GNN as well.

Definition 23. *For a set of classifiers $\Psi = \{\psi_1......\psi_m\}$, a Ψ-truth-table T is a 0/1 string of length m. T can be seen as a classifier, which classifies a node v to be true if and only if for any $1 \leq i \leq m$, the classification result of ψ_i on v equals to T_i, where T_i denotes the i-th bit of string T. We define $\mathcal{T}(\Psi) := \{0,1\}^m$ as the set of all Ψ-truth-tables. We have that for any graph G and its node v, v satisfies exactly one truth-table T.*

Proposition 24. *Let $\mathcal{FOC}_2(n)$ denote the set of formulas of $\mathcal{FOC}_2$ with quantifier depth no more than n. For any $\mathcal{G}_b$ and n, only finitely many intrinsically different node classifiers on $\mathcal{G}_b$ can be represented by $\mathcal{FOC}_2(n)$.*

Proof. Suppose all graphs in $\mathcal{G}_b$ have no more than N constants, then for any natural number $m > N$, formulas of form $\exists^{\geq m} y(\varphi(x,y))$ are always false. Therefore, it's sufficient only to consider $\mathcal{FOC}_2$ logical classifiers with threshold numbers no more than N on $\mathcal{G}_b$.

There are only finitely many predicates, and each boolean combination of unary predicates using $\wedge$, $\vee$,$\neg$ can be rewritten in the form of Disjunctive Normal Form (DNF)Davey and Priestley [2002]. So there are only finitely many intrinsically different formulas in $\mathcal{FOC}_2$ with quantifier depth 0.

By induction, suppose there are only finitely many intrinsically different $\mathcal{FOC}_2(k)$ formulas on $\mathcal{G}_b$. and each meaningful $\mathcal{FOC}_2(k+1)$ formula is generated by the following grammar

$$\varphi_1 \wedge \varphi_2,\varphi_1 \vee \varphi_2,\neg\varphi_2,\exists^{\geq m} y(\varphi'(x,y)),m \leq N \tag{18}$$

where φ_1,φ_2 are $\mathcal{FOC}_2(k+1)$ formulas and φ' is $\mathcal{FOC}_2(k)$ formulas.

Given that only the rule $\exists^{\geq n} y(\varphi'(x,y))$ can increase the quantifier depth from k to $k+1$, $m \leq N$, and there are only finitely many intrinsically different $\varphi'(x,y) \in \mathcal{FOC}_2(k)$ on $\mathcal{G}_b$ by induction. Therefore, there are only finitely many intrinsically different $\mathcal{FOC}_2(k+1)$ formulas of form $\exists^{\geq m} y(\varphi'(x,y))$ on $\mathcal{G}_b$. Moreover, their boolean combination using $\wedge$, $\vee$,$\neg$ can be always rewritten in the DNF form, So there are also finitely many intrinsically different $\mathcal{FOC}_2(k+1)$ logical classifiers on $\mathcal{G}_b$. $\square$

Lemma 25. *For any two pairs (G_1, v_1) and (G_2, v_2), where G_1 and G_2 are two* bounded *graphs from $\mathcal{G}_b$ and v_1 and v_2 are two nodes in G_1 and G_2, respectively. If all logical classifiers in $\mathcal{FOC}_2(L)$ can't distinguish v_1,v_2, then any R^2-GNN with layer no more than L can't distinguish them as well.*

Proof. By one-hot feature initialization function of R^2-GNN, $\mathcal{FOC}_2(0)$ can distinguish all different one-hot intial features, so the lemma trivially holds for the base case ($L = 0$).

For the inductive step, we suppose Lemma 25 holds for all $L \leq k$, then we can assume v_1,v_2 can't be distinguished by $\mathcal{FOC}_2(k+1)$. Let $N = k+1$

G_1 and G_2 are *bounded* graphs from $\mathcal{G}_b$, so $\mathcal{FOC}_2(N)$ has finitely many intrinsically different classifiers according to Proposition 24. Let $\mathcal{TT}_N(v)$ denote the $\mathcal{FOC}_2(N)$-truth-table satisfied by v. According to Definition 23, we know that for any $T \in \mathcal{T}(\mathcal{FOC}_2(N))$, there exists a $\mathcal{FOC}_2(N)$ classifier φ_T such that for any node v on G_i, where $i \in 1,2$, $\mathcal{TT}_N(v) = T \Leftrightarrow (G_i,v) \models \varphi_T$.

Assume there is an R^2-GNN $\mathcal{A}$ that distinguish v_1,v_2 with layer $L = k+1$. Let $\widehat{\mathcal{A}}$ denote its first k layers. By update rule of R^2-GNN illustrated in Equation 2, output of $\mathcal{A}$ on node v of graph G, $\mathbf{x}_v^{(k+1)}$ only dependent on the following three things:

- output of $\widehat{\mathcal{A}}$ on v, $\mathbf{x}_v^{(k)}$
- multiset of outputs of $\widehat{\mathcal{A}}$ on r-type neighbors of v for each $r \in P_2$, $\{\mathbf{x}_u^{(k)} | u \in \mathcal{N}_{G,r}(v)\}$
- multiset of outputs of $\widehat{\mathcal{A}}$ on all nodes in the graph, $\{\mathbf{x}_u^{(k)} | u \in \mathcal{N}_{G,r}(v)\}$

By induction, since v_1,v_2 can't be distinguished by $\mathcal{FOC}_2(k)$, they has same feature outputted by $\widehat{\mathcal{A}}$. Then there are two remaining possibilities.

- $\{\!\{\mathcal{TT}_k(u) | u \in \mathcal{N}_{G_1,r}(v_1)\}\!\} \neq \{\!\{\mathcal{TT}_k(u) | u \in \mathcal{N}_{G_2,r}(v_2)\}\!\}$ for some binary predicate r. Therefore, there exists a $\mathcal{FOC}_2(k)$-truth-table T, such that v_1,v_2 have differently many r-type neighbors that satisfies φ_T. Without loss of generality, suppose v_1,v_2 have n_1,$n_2(n_1 < n_2)$ such neighbors respectively. we can write a $\mathcal{FOC}_2(k+1)$ formula $\exists^{\geq n_2} y(r(x,y) \wedge \varphi_T(y))$ that distinguishes v_1 and v_2, which contradicts the precondition that they can't be distinguished by $\mathcal{FOC}_2(k+1)$ classifiers.
- $\{\!\{\mathcal{TT}_k(u) | u \in V(G_1)\}\!\} \neq \{\!\{\mathcal{TT}_k(u) | u \in V(G_2)\}\!\}$. Therefore, there exists a $\mathcal{FOC}_2(k)$-truth-table T, such that G_1,G_2 have differently many nodes that satisfies φ_T. Without loss of generality, suppose G_1,G_2 have n_1,$n_2(n_1 < n_2)$ such nodes respectively. we can write a $\mathcal{FOC}_2(k+1)$ formula $\exists^{\geq n_2} y \varphi_T(y)$ that distinguishes v_1 and v_2, which contradicts the precondition that they can't be distinguished by $\mathcal{FOC}_2(k+1)$ classifiers.

Since all possibilities contradicts the precondition that v_1,v_2 can't be distinguished by $\mathcal{FOC}_2(k+1)$, such an $\mathcal{A}$ that distinguishes v_1,v_2 doesn't exist. □

We can now gather all of these to prove Theorem 4.

Proof. For any R^2-GNN $\mathcal{A}$, suppose it has L layers. For any graph $G \in \mathcal{G}_b$ and its node v, let $\mathcal{TT}_L(v)$ denote the $\mathcal{FOC}_2(L)$-truth-table satisfied by v. For any $T \in \mathcal{T}(\mathcal{FOC}_2(L))$, since $\mathcal{G}_b$ is a bounded graph class, using Proposition 24, there exists a $\mathcal{FOC}_2(L)$ classifier φ_T such that for any node v in graph $G \in \mathcal{G}_b$, $\mathcal{TT}_L(v) = T \Leftrightarrow (G,v) \models \varphi_T$

By Lemma 25, If two nodes v_1,v_2 have same $\mathcal{FOC}_2(L)$-truth-table ($\mathcal{TT}_L(v_1) = \mathcal{TT}_L(v_2)$), they can't be distinguished by $\mathcal{A}$. Let S denote the subset of $\mathcal{T}(\mathcal{FOC}_2(L))$ that satisfies $\mathcal{A}$. By Proposition 24, $\Phi := \{\varphi_T | T \in S\}$ is a finite set, then disjunction of formulas in Φ, $(\bigvee_{T \in S} \varphi_T)$ is a $\mathcal{FOC}_2$ classifier that equals to $\mathcal{A}$ under bounded graph class $\mathcal{G}_b$. □

A.5 proof of Theorem 7

Theorem 7. *R^2-GNNs $\subseteq$ R^2-GNNs $\circ F$ on any* universal *graph class $\mathcal{G}_u$.*

Proof. Assume that we have a predicate set $P = P_1 \cup P_2$, $K = |P_2|$ and let $P' = P \cup \{primal,aux1,aux2\}$ denote the predicate set after transformation F. For any R^2-GNN $\mathcal{A}$ under P, we want to construct another R^2-GNN $\mathcal{A}'$ under P', such that for any graph G under P and its node v, v has the same feature outputted by $\mathcal{A}(G,v)$ and $\mathcal{A}'(F(G),v)$. Let L denote the layer number of $\mathcal{A}$.

We prove this theorem by induction over the number of layers L. In the base ($L = 0$), our result trivially holds since the one-hot initialization over P' contains all unary predicate information in P. Now suppose the result holds for $L \leq k$, so it suffices to prove it when $L = k + 1$.

For the transformed graph $F(G)$, *primal*(v) is *true* if and only if v is the node in the original graph G. Without loss of generality, if we use one-hot feature initialization on P', we can always keep an additional dimension in the node feature vector $\mathbf{x}_v$ to show whether *primal*(v) is *true*, its value is always $0/1$, in the proof below when we use $\mathbf{x}$ to denote the feature vectors, we omit this special dimension for simplicity. But keep in mind that this dimension always keeps so we can distinguish original nodes and added nodes.

Recall that an R^2-GNN is defined by $\{C^{(i)}, (A_j^{(i)})_{i=1}^K, R^{(i)}\}_{i=1}^L$. By induction, let $\widehat{\mathcal{A}}$ denote the first k layers of $\mathcal{A}$, and let $\widehat{\mathcal{A}}'$ denote the R^2-GNN equivalent with $\widehat{\mathcal{A}}$ on F transformation such that

$\widehat{\mathcal{A}} = \widehat{\mathcal{A}}' \circ F$. We will append three layers to $\widehat{\mathcal{A}}'$ to construct $\mathcal{A}'$ that is equivalent to $\mathcal{A}$. Without loss of generality, we can assume all layers in $\mathcal{A}$ have same dimension length d. Suppose L' is the layer number of $\widehat{\mathcal{A}}'$, so we will append layer $L'+1$,$L'+2$,$L'+3$. for all $l \in \{L'+1,L'+2,L'+3\}$, let $\{C^{a,(l)},C^{p,(l)},(A_j^{*,(l)})_{j=1}^K,A_{aux1}^{*,(l)},A_{aux2}^{*,(l)},R^{*,(l)}\}$ denote the parameters in l-th layer of $\mathcal{A}$. Here, $A_{aux1}^{*,(l)},A_{aux2}^{*,(l)}$ denotes the aggregation function corresponding to two new predicates *aux1*,*aux2*, added in transformation F, and $C^{p,(l)},C^{a,(l)}$ are different combination function that used for primal nodes and non-primal nodes. Note that with the help of the special dimension mentioned above, we can distinguish primal nodes and non-primal nodes. Therefore, It's safe to use different combination functions for these two kinds of nodes. Note that here since we add two predicates *aux1*,*aux2*, the input for combination function should be in the form $C^p(\mathbf{x}_0,(\mathbf{x}_j)_{j=1}^K,\mathbf{x}_{aux1},\mathbf{x}_{aux2},\mathbf{x}_g)$ where $\mathbf{x}_0$ is the feature vector of the former layer, and $\mathbf{x}_j,1 \le j \le K$ denote the output of aggregation function $A_j^{*,(l)}$, $\mathbf{x}_{aux1},\mathbf{x}_{aux2}$ denote the output of aggregation function $A_{aux1}^{*,(l)},A_{aux2}^{*,(l)}$, and $\mathbf{x}_g$ denotes the feature outputted by global readout function $R^{*,(l)}$. For aggregation function and global readout function, their inputs are denoted by $\mathbf{X}$, meaning a multiset of feature vector. Note that all aggregation functions and readout functions won't change the feature dimension, only combination functions $C^{p,(l)},C^{a,(l)}$ will transform d_{l-1} dimension features to d_l dimension features.

1). layer $L'+1$: input dimension is d, output dimension is $d' = Kd$. For feature vector $\mathbf{x}$ with length d', let $\mathbf{x}^{(i)}, i \in \{1,\dots,K\}$ denote its i-th slice in dimension $[(i-1)d+1,id]$. Let $[\mathbf{x}_1,\dots,\mathbf{x}_m]$ denote concatenation of $\mathbf{x}_1,\dots,\mathbf{x}_m$, and let $[\mathbf{x}]^n$ denote concatenation of n copies of $\mathbf{x}$, $\mathbf{0}^n$ denote zero vectors of length n. parameters for this layer are defined below:

$$C^{p,(L'+1)}(\mathbf{x}_0,(\mathbf{x}_j)_{j=1}^K,\mathbf{x}_{aux1},\mathbf{x}_{aux2},\mathbf{x}_g) = [\mathbf{x}_0,\mathbf{0}^{d'-d}] \tag{19}$$

$$C^{a,(L'+1)}(\mathbf{x}_0,(\mathbf{x}_j)_{j=1}^K,\mathbf{x}_{aux1},\mathbf{x}_{aux2},\mathbf{x}_g) = [\mathbf{x}_{aux1}]^K \tag{20}$$

$$A_{aux1}^{*,(L'+1)}(\mathbf{X}) = \sum_{\mathbf{x}\in\mathbf{X}} \mathbf{x} \tag{21}$$

Other parameters in this layer are set to functions that always output zero-vector.

We can see here that the layer $L'+1$ do the following thing:

For all primal nodes a and its non-primal neighbor e_{ab}, pass concatenation of K copies of $\mathbf{x}_a$ to $\mathbf{x}_{e_{ab}}$, and remains the feature of primal nodes unchanged.

2). layer $L'+2$, also has dimension $d' = Kd$, has following parameters.

$$C^{p,(L'+2)}(\mathbf{x}_0,(\mathbf{x}_j)_{j=1}^K,\mathbf{x}_{aux1},\mathbf{x}_{aux2},\mathbf{x}_g) = \mathbf{x}_0 \tag{22}$$

$$C^{a,(L'+2)}(\mathbf{x}_0,(\mathbf{x}_j)_{j=1}^K,\mathbf{x}_{aux1},\mathbf{x}_{aux2},\mathbf{x}_g) = \sum_{j=1}^K \mathbf{x}_j \tag{23}$$

$$\forall j \in [1,K], A_j^{*,(L'+2)}(\mathbf{X}) = [\mathbf{0}^{(j-1)d}, \sum_{\mathbf{x}\in\mathbf{X}} \mathbf{x}^{(j)}, \mathbf{0}^{(K-j)d}] \tag{24}$$

All other parameters in this layer are set to function that always outputs zero vectors. This layer do the following thing:

For all primal nodes, keep the feature unchanged, for all added node pair e_{ab},e_{ba}. Switch their feature, but for all $r_i \in P_2$, if there is no r_i relation between a,b, the i-th slice of $\mathbf{x}_{e_{ab}}$ and $\mathbf{x}_{e_{ba}}$ will be set to $\mathbf{0}$.

3). layer $L'+3$, has dimension d, and following parameters.

$$C^{p,(L'+3)}(\mathbf{x}_0,(\mathbf{x}_j)_{j=1}^K,\mathbf{x}_{aux1},\mathbf{x}_{aux2},\mathbf{x}_g) = C^{(L)}(\mathbf{x}_0^{(1)},(\mathbf{x}_{aux1}^{(j)})_{j=1}^K,\mathbf{x}_g^{(1)}) \tag{25}$$

$$R^{*,(L'+3)}(\mathbf{X}) = [R^{(L)}(\{\!\{\mathbf{x}_v^{(1)} | \mathbf{x}_v \in \mathbf{X}, \mathbf{primal}(v)\}\!\}), \mathbf{0}^{d'-d}] \tag{26}$$

$$A_{aux1}^{*,(L'+3)}(\mathbf{X}) = [A_1^{(L)}(\{\!\{\mathbf{x}^{(1)} | \mathbf{x} \in \mathbf{X}\}\!\}) \dots\dots A_K^{(L)}(\{\!\{\mathbf{x}^{(K)} | \mathbf{x} \in \mathbf{X}\}\!\})] \tag{27}$$

Note that $C^{(L)},A_j^{(L)},R^{(L)}$ are all parameters in the last layer of $\mathcal{A}$ mentioned previously. All other parameters in this layer are set to functions that always output zero vectors. We can see that this layer simulates the work of last layer of $\mathcal{A}$ as follows:

- For all $1 \le j \le K$, use the j-th slice of feature vector $\mathbf{x}^{(j)}$ to simulate $A_j^{(L)}$ and store results of aggregation function $A_j^{(L)}$ on this slice.
- Global readout trivially emulates what $R^{(L)}$ does, but only reads features for primal nodes. It can be done since we always have a special dimension in feature to say whether it's a primal node.
- We just simulate what $C^{(L)}$ does on primal nodes. For $1 \le j \le K$ The type r_j aggregation result (output of $A_j^{(L)}$) used for input of $C^{(L)}$ is exactly j-th slice of return value of $A_{aux1}^{*,(L'+3)}$.

By construction above, $\mathcal{A}'$ is a desired model that have the same output as $\mathcal{A}$.

□

A.6 proof of Theorem 8

Theorem 8. *$\mathcal{FOC}_2 \subseteq$ R^2-GNNs $\circ F$ on any* universal *graph class $\mathcal{G}_u$.*

Proof. For any $\mathcal{FOC}_2$ classifier φ under predicate set P, we want to construct a 0/1-GNN $\mathcal{A}$ on $P' = P \cup \{primal, aux1, aux2\}$ equivalent to φ with *graph transformation F*.

Recall that $\mathcal{FOC}_2 = \mathcal{RSFOC}_2$ shown in Lemma 20 and 0/1-GNNs $\subseteq$ R^2-GNNs, it suffices to prove that 0/1-GNN$\circ F$ capture $\mathcal{RSFOC}_2$. By Lemma 17, since $\wedge, \vee, \neg$ are closed under 0/1-GNN it suffices to show that when φ is in the form $\exists^{\ge n}\big(\varphi_S(x,y) \wedge \varphi'(y)\big), S \subseteq P_2$, we can capture it.

We prove by induction over quantifier depth m of φ. Since 0-depth formulas are only about unary predicate that can be extracted from one-hot initial feature, our theorem trivially holds for $m = 0$. Now, we assume it also holds for $m \le k$, it suffices to prove the case when $m = k + 1$. Then there are two possibilities:

1. When $S \neq \emptyset$:

Consider the following logical classifier under P':

$$\widehat{\varphi}_S(x) := \Big(\bigwedge_{r\in S} \exists x r(x,y)\Big) \wedge \Big(\bigwedge_{r\notin S} \neg\exists x r(x,y)\Big) \tag{28}$$

$\widehat{\varphi}_S(x)$ restricts that for any $r \in P'$, x has r-type neighbor if and only if $r \in S$. Review the definition of transformation F, we know that for any added node e_{ab}, $(F(G), e_{ab}) \models \widehat{\varphi}_S$ if and only if $(G, a, b) \models \varphi_S(a, b)$, where $\varphi_S(x, y)$ is the *relation-specification* formula defined in Definition 19 That is to say for any $r_i, 1 \le i \le K$, there is relation r_i between a,b if and only if $i \in S$.

Now consider the following formula:

$$\widehat{\varphi} := \exists^{\ge n} y\Big(\textit{aux1}(x,y) \wedge \widehat{\varphi}_S(y) \wedge \Big(\exists x\big(\textit{aux2}(x,y) \wedge (\exists y(\textit{aux1}(x,y) \wedge \varphi'(y)))\big)\Big)\Big) \tag{29}$$

For any graph G and its node v, it's easy to see that $(G, v) \models \varphi \Leftrightarrow (F(G), v) \models \widehat{\varphi}$. Therefore we only need to capture $\widehat{\varphi}$ by 0/1-GNN on every primal node of transformed graphs. By induction, since quantifier depth of $\varphi'(y)$ is no more than k, we know $\varphi'(y)$ is in 0/1-GNN. $\widehat{\varphi}$ is generated from $\varphi'(y)$ using rules $\wedge$ and $\exists y\big(r(x,y) \wedge \varphi'(y)\big)$. By Lemma 17, $\wedge$ is closed under 0/1-GNN. For $\exists y\big(r(x,y) \wedge \varphi'(y)\big)$, we find that the construction needed is the same as construction for single-element S on single-edge graph class $\mathcal{G}_s$ used in Theorem 3. Therefore, since we can manage these two rules, we can also finish the construction for $\widehat{\varphi}$, which is equivalent to φ on primal nodes of transformed graph.

2. When $S = \emptyset$

First, consider the following two logical classifiers:

$$\bar{\varphi}(x) := \Big(\textit{primal}(x) \wedge \varphi'(x)\Big) \tag{30}$$

$\bar{\varphi}$ says a node is primal, and satisfies $\varphi'(x)$. Since $\varphi'(x)$ has quantifier depth no more than k, and $\wedge$ is closed under 0/1-GNN. There is a 0/1-GNN $\mathcal{A}_1$ equivalent to $\bar{\varphi}$ on transformed graph. Then, consider the following formula.

$$\tilde{\varphi}(x) := \exists y\big(aux2(x,y) \wedge (\exists x, aux1(x,y) \wedge \varphi'(x))\big) \tag{31}$$

$\tilde{\varphi}(x)$ evaluates on added nodes e_{ab} on transformed graph, e_{ab} satisfies it iff b satisfies φ'

Now for a graph G and its node v, define n_1 as the number of nodes on $F(G)$ that satisfies $\bar{\varphi}$, and define n_2 as the number of *aux1*-type neighbors of v on $F(G)$ that satisfies $\tilde{\varphi}$. Since $\varphi(x) = \exists^{\geq n} y(\varphi_\emptyset(x,y) \wedge \varphi'(y))$ It's easy to see that $(G,v) \models \varphi$ if and only if $n_1 - n_2 \geq n$.

Formally speaking, for a node set S, let $|S|$ denote number of nodes in S, we define the following classifier c such that for any graph G and its node a, $c(F(G),a) = 1 \Leftrightarrow (G,a) \models \varphi$

$$c(F(G),a) = 1 \Leftrightarrow |\{v|v \in V(F(G)), (F(G),v) \models \bar{\varphi}\}| - |\{v|v \in \mathcal{N}_{F(G),aux1}(v), (F(G),v) \models \tilde{\varphi}\}| \geq n \tag{32}$$

So how to construct a model $\mathcal{A}$ to capture classifier c? First, by induction $\bar{\varphi}$,$\tilde{\varphi}$ are all formulas with quantifier depth no more than k so by previous argument there are 0/1-GNN models $\bar{\mathcal{A}}$,$\tilde{\mathcal{A}}$ that capture them respectively. Then we can use feature concatenation technic introduced in Equation (6) to construct a model $\widehat{\mathcal{A}}$ based on $\bar{\mathcal{A}}$,$\tilde{\mathcal{A}}$, such that $\widehat{\mathcal{A}}$ has two-dimensional output, whose first and second dimensions have the same output as $\bar{\mathcal{A}}$,$\tilde{\mathcal{A}}$ respectively.

Then, suppose $\widehat{\mathcal{A}}$ has L layers, The only thing we need to do is to append a new layer $L+1$ to $\widehat{\mathcal{A}}$, it has output dimension 1. parameters of it are $\{C^{(L+1)}, (A_j^{(L+1)})_{j=1}^K, A_{aux1}^{(L+1)}, A_{aux2}^{(L+1)}, R^{(L+1)}\}$ as defined in Equation (5). The parameter settings are as follows:

$\mathbf{R}_{1,1}^{(L+1)} = 1, \mathbf{A}_{aux1,(1,2)}^{(L+1)} = -1, \mathbf{b}_1^{(L+1)} = 1-n$. Other parameters are set to 0, where $\mathbf{A}_{aux1,(1,2)}^{(L+1)}$ denotes the value in the first row and second column of $\mathbf{A}_{aux1}^{(L+1)}$.

In this construction, we have

$\mathbf{x}_v^{(L+1)} = max(0, min(1, \sum_{u\in V(F(G))} \mathbf{x}_{u,1}^{(L)} - \sum_{u\in\mathcal{N}_{F(G),aux1}(v)} \mathbf{x}_{u,2}^{(L)} - (n-1)))$, which has exactly the same output as classifier c defined above in Equation (32). Therefore, $\mathcal{A}$ is a desired model. $\square$

A.7 proof of Theorem 9

Theorem 9. *R^2-GNNs $\circ F \subseteq \mathcal{FOC}_2$ on any* bounded *graph class $\mathcal{G}_b$.*

Before we go into theorem itself, we first introduce Lemma 26 that will be used in following proof.

Lemma 26. *Let $\varphi(x,y)$ denote a $\mathcal{FOC}_2$ formula with two free variables, for any natural number n, the following sentence can be captured by $\mathcal{FOC}_2$:*

***There exists no less than** n **ordered node pairs** (a,b) **such that** $(G,a,b) \models \varphi$.*

Let c denote the graph classifier such that $c(G) = 1$ iff G satisfies the sentence above.

Proof. The basic intuition is to define $m_i, 1 \leq i < n$ as the number of nodes a, such that there are **exactly** i nodes b that $\varphi(a,b)$ is *true*. Specially, we define m_n as the number of nodes a, such that there are **at least** n nodes b that $\varphi(a,b)$ is *true*. Since $\sum_{i=1}^n i m_i$ exactly counts the number of valid ordered pairs when $m_n = 0$, and it guarantees the existence of at least n valid ordered pairs when $m_n > 0$. It's not hard to see that for any graph G, $c(G) = 1 \Leftrightarrow \sum_{i=1}^n i m_i \geq n$. Futhermore, fix a valid sequence $(m_1......m_n)$ such that $\sum_{i=1}^n i m_i \geq n$, there has to be another sequence $(k_1......k_n)$ such that $n \leq \sum_{i=1}^n i k_i \leq 2n$ and $k_i \leq m_i$ for all $1 \leq i \leq n$. Therefore, We can enumerate all possibilities of valid $(k_1......k_n)$, and for each valid $(k_1......k_n)$ sequence, we judge whether there are **at least** k_i such nodes a for every $1 \leq i \leq n$.

Formally, $\varphi_i(x) := \exists^{[i]} y \varphi(x,y)$ can judge whether a node a has exactly i partners b such that $\varphi(a,b) = 1$, where $\exists^{[i]} y \varphi(x,y)$ denotes "there are exactly i such nodes y" which is the abbreviation of formula $(\exists^{\geq i} y \varphi(x,y)) \wedge (\neg \exists^{\geq i+1} y \varphi(x,y))$. The $\mathcal{FOC}_2$ formula equivalent to our desired sentence

c is as follows:

$$\bigvee_{\sum_{i=1}^{n} n \leq ik_i \leq 2n} \left(\bigwedge_{i=1}^{n-1} \exists^{\geq k_i} x \left(\exists^{[i]} y \varphi(x,y) \right) \right) \wedge \left(\exists^{\geq k_n} x \left(\exists^{\geq n} y \varphi(x,y) \right) \right) \tag{33}$$

This $\mathcal{FOC}_2$ formula is equivalent to our desired classifier c. $\square$

With the Lemma 26, we now start to prove Theorem 9.

Proof. By Theorem 4, it follows that R^2-GNNs $\circ F \subseteq \mathcal{FOC}_2 \circ F$. Therefore it suffices to show $\mathcal{FOC}_2 \circ F \subseteq \mathcal{FOC}_2$.

By Lemma 20, it suffices to show $\mathcal{RSFOC}_2 \circ F \subseteq \mathcal{FOC}_2$. Since $\wedge, \vee, \neg$ are common rules. We only need to show for any $\mathcal{RSFOC}_2$ formula of form $\varphi(x) := \exists^{\geq n} y(\varphi_S(x,y) \wedge \varphi'(y))$ under transformed predicate set $P' = P \cup \{aux1, aux2, primal\}$, there exists an $\mathcal{FOC}_2$ formula φ^1 such that for any graph G under P and its node v, $(G,v) \models \varphi^1 \Leftrightarrow (F(G),v) \models \varphi$.

In order to show this, we consider a stronger result:

For any such formula φ, including the existence of valid φ^1, we claim there also exists an $\mathcal{FOC}_2$ formula φ^2 with two free variables such that the following holds: for any graph G under P and its added node e_{ab} on $F(G)$, $(G,a,b) \models \varphi^2 \Leftrightarrow (F(G),e_{ab}) \models \varphi$. Call φ^1,φ^2 as first/second discriminant of φ.

Now we need to prove the existence of φ^1 and φ^2.

We prove by induction over quantifier depth m of φ, Since we only add a single unary predicate *primal* in P', any $\varphi(x)$ with quantifier depth 0 can be rewritten as $(primal(x) \wedge \varphi^1(x)) \vee (\neg primal(x) \wedge \varphi^2(x))$, where $\varphi^1(x),\varphi^2(x)$ are two formulas that only contain predicates in P. Therefore, φ^1,φ^2 can also be seen as first/second discriminat of φ, so our theorem trivially holds for $m = 0$. Now assume it holds for $m \leq k$, we can assume quantifier depth of $\varphi = \exists^{\geq n} y(\varphi_S(x,y) \wedge \varphi'(y))$ is $m = k+1$.

Consider the construction rules of transformation F, for any two primal nodes in $F(G)$, there is no relation between them, for a primal node a and an added node e_{ab}, there is exactly a single relation of type *aux1* between them. For a pair of added nodes e_{ab},e_{ba}, there are a bunch of relations from the original graph G and an additional *aux2* relation between them. Therefore, it suffices to only consider three possible kinds of $S \subseteq P_2 \cup \{aux1, aux2\}$ according to three cases mentiond above. Then, we will construct first/second determinants for each of these three cases. Since $\varphi'(y)$ has quantifier depth no more than k, by induction let $\widehat{\varphi}^1,\widehat{\varphi}^2$ be first/second discriminants of φ' by induction.

1. $S = \{\textbf{aux1}\}$:

for primal node a, $\varphi(a)$ means the following: there exists at least n nodes b, such that there is some relation between a,b on G and the added node e_{ab} on $F(G)$ satisfies φ'. Therefore, the first determinant of φ can be defined as following:

$$\varphi^1(x) := \exists^{\geq n} y, \Big(\bigvee_{r \in P_2} r(x,y) \Big) \wedge \widehat{\varphi}^2(x,y) \tag{34}$$

for added nodes e_{ab} on $F(G)$, $\varphi(e_{ab})$ means a satisfies φ', so the second determinant of φ is the following:

$$n = 1 : \varphi^2(x,y) := \widehat{\varphi}^1(x), \;\; n > 1 : \varphi^2(x,y) := \bot \tag{35}$$

2.$S = \{\textbf{aux2}\} \cup T, T \subseteq P_2, T \neq \emptyset$

primal nodes don't have *aux2* neighbors, so first determinant is trivially *false*.

$$\varphi^1(x) := \bot \tag{36}$$

For added node e_{ab}, e_{ab} satisfies φ iff there are exactly relations between a,b of types in T, and e_{ba} satisfies φ'. Therefore the second determinant is as follows, where $\varphi_T(x,y)$ is the *relation-specification* formula under P introduced in Definition 19

$$n = 1 : \varphi^2(x,y) := \varphi_T(x,y) \wedge \widehat{\varphi}^2(y,x), \, n > 1 : \varphi^2(x,y) := \bot \tag{37}$$

3. $S = \emptyset$

For a subset $S \subseteq P_2 \cup \{aux1,aux2\}$, let $\varphi_S(x,y)$ denote the *relation-specification* formula under $P_2 \cup \{aux1,aux2\}$ defined in Definition 19.

Since we consider on bounded graph class $\mathcal{G}_b$, node number is bounded by a natural number N. For any node a on $F(G)$, let m denote the number of nodes b on $F(G)$ such that $\varphi'(b) = 1$, let m_0 denote the number of nodes b on $F(G)$ such that $\varphi'(b) = 1$ and there is a single relation *aux1*, between (a,b) on $F(G)$, (That is equivalent to $\varphi_{\{aux1\}}(a,b) = 1$). For any $T \subseteq P_2$, let m_T denote the number of nodes b on $F(G)$ such that $\varphi'(b) = 1$ and a,b has exactly relations of types in $T \cup \{aux2\}$ on $F(G)$, (That is equivalent to $\varphi_{T\cup\{aux2\}}(a,b) = 1$).

Note that the number of nodes b on $F(G)$ such that a,b don't have any relation, (That is equivalent to $\varphi_{\emptyset}(a,b) = 1$) and $\varphi'(b) = 1$ equals to $m - m_0 - \sum_{T\subseteq P_2} m_T$. Therefore, for any transformed graph $F(G)$ and its node v, $(F(G),v) \models \varphi \Leftrightarrow m - m_0 - \sum_{T\subseteq P_2} m_T \geq n$. Since $|V(G)| \leq N$ for all G in bounded graph class $\mathcal{G}_b$, transformed graph $F(G)$ has node number no more than N^2. Therefore, we can enumerate all possibilities of $m,m_0,m_T \leq N^2, T \subset P_2$ such that the above inequality holds, and for each possibility, we judge whehter there exists exactly such number of nodes for each corresponding parameter. Formally speaking, φ can be rewritten as the following form:

$$\tilde{\varphi}_{m,m_0}(x) := \left(\exists^{[m]} y \varphi'(y)\right) \wedge \left(\exists^{[m_0]} y(\varphi_{\{aux1\}}(x,y) \wedge \varphi'(y))\right)) \tag{38}$$

$$\varphi(x) \equiv \bigvee_{m-m_0-\sum_{T\subseteq P_2} \geq n, 0\leq m,m_0,m_T \leq N^2} \left(\tilde{\varphi}_{m,m_0}(x) \wedge \Big(\bigwedge_{T\subseteq P_2} \exists^{[m_T]} y, (\varphi_{T\cup\{aux2\}}(x,y) \wedge \varphi'(y))\Big)\right) \tag{39}$$

where $\exists^{[m]} y$ denotes there are exactly m nodes y.

Since first/second determinant can be constructed trivially under combination of $\wedge$, $\vee$,$\neg$, and we've shown how to construct determinants for formulas of form $\exists^{\geq n} y(\varphi_S(x,y) \wedge \varphi'(y))$ when $S = \{aux1\}$ and $S = \{aux2\} \cup T, T \subseteq P_2$ in the previous two cases. Therefore, in Equation (38) and Equation (39), the only left part is the formula of form $\exists^{[m]} y \varphi'(y)$. The only remaining work is to show how to construct first/second determinants for formula in form $\varphi(x) := \exists^{\geq n} y \varphi'(y)$.

Let m_1 denote the number of primal nodes y that satisfies $\varphi'(y)$ and let m_2 denote the number of non-primal nodes y that satisfies $\varphi'(y)$. It's not hard to see that for any node v on $F(G)$, $(F(G),v) \models \varphi \Leftrightarrow m_1 + m_2 \geq n$. Therefore, $\varphi(x) = \exists^{\geq n} y \varphi'(y)$ that evaluates on $F(G)$ is equivalent to the following sentence that evaluates on G: *"There exists two natural numbers m_1,m_2 such that the following conditions hold:* ***1.*** *$m_1 + m_2 = n$.* ***2.*** *There are at least m_1 nodes b on G that satisfies $\widehat{\varphi}^1$, (equivalent to $(F(G),b) \models \varphi'$).* ***3.*** *There are at least m_2 ordered node pairs a,b on G such that a,b has some relation and $(G,a,b) \models \widehat{\varphi}^2$, (equivalent to $(F(G),e_{ab}) \models \varphi'$)."*

Formally speaking, rewrite the sentence above as formula under P, we get the following construction for first/second determinants of φ.

$$\varphi^1(x) = \varphi^2(x,y) = \bigvee_{m_1+m_2=n} \left((\exists^{\geq m_1} y, \widehat{\varphi}^1(y)) \wedge \overline{\varphi}_{m_2} \right) \tag{40}$$

where $\overline{\varphi}_{m_2}$ is the $\mathcal{FOC}_2$ formula that expresses *"There exists at least m_2 ordered node pairs (a,b) such that $(G,a,b) \models \widehat{\varphi}^2(x,y) \wedge (\bigvee_{r\in P_2} r(x,y))$"*. We've shown the existence of $\overline{\varphi_{m_2}}$ in Lemma 26 $\square$

A.8 Proof of Theorem 13

Theorem 13. *time-and-graph* $\subsetneq$ *R*^{2}*-TGNN* $\circ F^T =$ *time-then-graph.*

For a graph G with n nodes, let $\mathbb{H}^V \in \mathbb{R}^{n\times d_v}$ denote node feature matrix, and $\mathbb{H}^E \in \mathbb{R}^{n\times n\times d_e}$ denote edge feature matrix, where $\mathbb{H}^E_{ij}$ denote the edge feature vector from i to j.

First we need to define the GNN used in their frameworks. Note that for the comparison fairness, we add the the global readout to the node feature update as we do in R^2-GNNs. It recursively calculates the feature vector $\mathbb{H}^{V,(l)}_i$ of the node i at each layer $1 \leq l \leq L$ as follows:

$$\mathbb{H}^{V,(l)}_i = u^{(l)}\left(g^{(l)}(\{\{(\mathbb{H}^{V,(l-1)}_i, \mathbb{H}^{V,(l-1)}_j, \mathbb{H}^E_{ij}) \mid j \in \mathcal{N}(i)\}\}), r^{(l)}(\{\{\mathbb{H}^{V,(l-1)}_j | j \in V\}\})\right) \tag{41}$$

where $\mathcal{N}(i)$ denotes the set of all nodes that adjacent to i, and $u^{(l)}$,$g^{(l)}$,$r^{(l)}$ are learnable functions. Note that here the GNN framework is a little different from the general definition defined in Equation (2). However, this framework is hard to fully implement and many previous works implementing *time-and-graph* or *time-then-graph* Gao and Ribeiro [2022] (Li et al. [2019], Seo et al. [2016], Chen et al. [2018], Manessi et al. [2020], Sankar et al. [2018],Rossi et al. [2020b]) don't reach the expressiveness of Equation (41). This definition is more for the theoretical analysis. In contrast, our definition for GNN in Equation (1) and Equation (2) is more practical since it is fully captured by a bunch of commonly used models such as Schlichtkrull et al. [2018]. For notation simplicity, for a GNN $\mathcal{A}$, let $\mathbb{H}^{V,(L)} = \mathcal{A}(\mathbb{H}^V,\mathbb{H}^E)$ denote the node feature outputted by $\mathcal{A}$ using $\mathbb{H}^V$,$\mathbb{H}^E$ as initial features.

Proposition 27. *(Gao and Ribeiro [2022]):time-and-graph $\subsetneq$ time-then-grahp*

The above proposition is from **Theorem 1** of Gao and Ribeiro [2022]. Therefore, in order to complete the proof of Theorem 13, we only need to prove R^2-TGNN $\circ F^T$ = *time-then-graph*.

Let $G = \{G_1, \ldots, G_T\}$ denote a temporal knowledge graph, and $\mathbb{A}^t \in \mathbb{R}^{n\times|P_1|}$, $\mathbb{E}^t \in \mathbb{R}^{n\times n\times|P_2|}$, $1 \leq t \leq T$ denonte one-hot encoding feature of unary facts and binary facts on timestamp t, where P_1,P_2 are unary and binary predicate sets.

The updating rule of a *time-then-graph* model can be generalized as follows:

$$\forall i \in V,\ \mathbb{H}_i^V = \textbf{RNN}([\mathbb{A}_i^1......\mathbb{A}_i^T]) \tag{42}$$

$$\forall i,j \in V,\ \mathbb{H}_{i,j}^E = \textbf{RNN}([\mathbb{E}_{i,j}^1......\mathbb{E}_{i,j}^T]) \tag{43}$$

$$\mathbf{X} := \mathcal{A}(\mathbb{H}^V,\mathbb{H}^E) \tag{44}$$

where $\mathcal{A}$ is a GNN defined above, **RNN** is an arbitrary Recurrent Neural Network. $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the final node feature output of *time-then-graph*.

First we need to prove *time-then-graph* $\subseteq$ R^2-TGNN$\circ F^T$. That is, for any *time-then-graph* model, we want to construct an equivalent R^2-TGNN $\mathcal{A}'$ to capture it on transformed graph. We can use nodes added after transformation to store the edge feature $\mathbb{H}^E$, and use primal nodes to store the node feature $\mathbb{H}^V$. By simulating **RNN** through choosing specific functions in R^2-TGNN, we can easily construct a R^2-TGNN $\mathcal{A}'$ such that for any node i, and any node pair i,j with at least one edge in history, $\mathbf{x}_i = \mathbb{H}_i^V$ and $\mathbf{x}_{e_{ij}} = \mathbb{H}_{i,j}^E$ hold, where $\mathbf{x}_i$ and $\mathbf{x}_{e_{ij}}$ are features of corresponding primal node i and added node e_{ij} outputted by $\mathcal{A}'$.

Note that $\mathcal{A}'$ is a R^2-TGNN, it can be represented as $\mathcal{A}'_1......\mathcal{A}'_T$, where each $\mathcal{A}'_t, 1 \leq t \leq T$ is a R^2-GNN. $\mathcal{A}'$ has simulated work of **RNN**, so the remaining work is to simulate $\mathcal{A}(\mathbb{H}^V,\mathbb{H}^E)$. We do the simulation over induction on layer number L of $\mathcal{A}$.

When $L = 0$, output of $\mathcal{A}$ is exactly $\mathbb{H}^V$, which has been simulated by $\mathcal{A}'$ above.

Suppose $L = k + 1$, let $\tilde{\mathcal{A}}$ denote R^2-GNN extracted from $\mathcal{A}$ but without the last layer $k + 1$. By induction, we can construct a R^2-TGNN $\tilde{\mathcal{A}}'$ that simulates $\tilde{\mathcal{A}}(\mathbb{H}^V,\mathbb{H}^E)$. Then we need to append three layers to $\tilde{\mathcal{A}}'$ to simulate the last layer of $\mathcal{A}$.

Let $u^{(L)}$,$g^{(L)}$,$r^{(L)}$ denote parameters of the last layer of $\mathcal{A}$. Using notations in Equation (2), let $\{C^{(l)},(A_j^{(l)})_{j=1}^{|P_2|},A_{aux1}^{(l)},A_{aux2}^{(l)},R^{(l)}\}_{l=1}^{3}$ denote parameters of the three layers appended to $\tilde{\mathcal{A}}'_T$. They are defined as follows:

First, we can choose specific function in the first two added layers, such that the following holds:

1. For any added node e_{ij}, feature outputted by the new model is $\mathbf{x}_{e_{ij}}^{(2)} = [\mathbb{H}_{ij}^E,\mathbf{x}'_i,\mathbf{x}'_j]$, where $\mathbf{x}^{(2)}$ denotes the feature outputted by the second added layer, and $\mathbf{x}'_i$,$\mathbf{x}'_j$ are node features of i,j outputted by $\tilde{\mathcal{A}}'$. For a feature $\mathbf{x}$ of added node of this form, we define $\mathbf{x}_0$,$\mathbf{x}_1$,$\mathbf{x}_2$ as corresponding feature slices where $\mathbb{H}_{ij}^E$,$\mathbf{x}'_i$,$\mathbf{x}'_j$ have been stored.

2. For any primal node, its feature $\mathbf{x}$ only stores $\mathbf{x}'_i$ in $\mathbf{x}_1$, and $\mathbf{x}_0$,$\mathbf{x}_2$ are all slices of dummy bits.

Let $\mathbf{X}$ be a multiset of features that represents function input. For the last added layer, we can choose specific functions as follows:

$$R^{(3)}(\mathbf{X}) := r^{(L)}(\{\!\{\mathbf{x}_1 | \mathbf{x} \in \mathbf{X}, \mathbf{primal}(\mathbf{x})\}\!\}) \tag{45}$$

$$A^{(3)}_{aux1}(\mathbf{X}) := g^{(L)}(\{\!\{(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_0) | \mathbf{x} \in \mathbf{X}\}\!\}) \tag{46}$$

$$C^{(3)}(\mathbf{x}_{aux1}, \mathbf{x}_g) := u^{(L)}(\mathbf{x}_{aux1}, \mathbf{x}_g) \tag{47}$$

where $\mathbf{x}_{aux1}, \mathbf{x}_g$ are outputs of $R^{(3)}$ and $A^{(3)}_{aux1}$, and all useless inputs of $C^{(3)}$ are omitted. Comparing this construction with Equation (41). It's east to see that after the last layer appended, we can construct an equivalent R^2-TGNN $\mathcal{A}'$ that captures $\mathcal{A}$ on transformed graph. By inductive argument, we prove *time-then-graph* $\subseteq$ R ^{2}-TGNN $\circ F^T$.

Then we need to show R^2-TGNN $\circ F^T \subseteq$ *time-then-graph*.

In Theorem 14, we will prove R^2-TGNN $\circ F^T =$ R^2-GNN $\circ F \circ H$. Its proof doesn't dependent on Theorem 13, so let's assume it's true for now. Then, instead of proving R^2-TGNN $\circ F^T$, it's sufficient to show R^2-GNN $\circ F \circ H \subseteq$ *time-then-graph*.

Let P_1^T, P_2^T denote the set of temporalized unary and binary predicate sets defined in Definition 10. Based on *most expressive ability* of Recurrent Neural Networks shown in Siegelmann and Sontag [1992], we can get a *most expressive representation* for unary and binary fact sequences through **RNN**. A *most expressive* RNN representation function is always injective, thus there exists a decoder function translating most-expressive representations back to raw sequences. Therefore, we are able to find an appropriate **RNN** such that its output features $\mathbb{H}^V, \mathbb{H}^E$ in Equation (42), Equation (43) contain all information needed to reconstruct all temporalized unary and binary facts related to the corresponding nodes.

For any R^2-GNN $\mathcal{A}$ on transformed collpsed temporal knowledge graph, we want to construct an equivalent *time-then-graph* model $\{\mathbf{RNN}, \mathcal{A}'\}$ to capture $\mathcal{A}$. In order to show the existence of the *time-then-graph* model, we will do an inductive construction over layer number L of $\mathcal{A}$. Here in order to build inductive argument, we will consider a following stronger result and aim to prove it: In additional to the existence of $\mathcal{A}'$, we claim there also exists a function $f_{\mathcal{A}}$ with the following property: For any two nodes a,b with at least one edge, $f_{\mathcal{A}}(\mathbf{x}'_a, \mathbf{x}'_b, \mathbb{H}^E_{ab}) = \mathbf{x}_{e_{ab}}$, where $\mathbf{x}'_a, \mathbf{x}'_b, \mathbb{H}^E_{ab}$ are features of a, b and edge information between a,b outputted by $\mathcal{A}'$, and $\mathbf{x}_{e_{ab}}$ is the feature of added node e_{ab} outputted by $\mathcal{A} \circ F \circ H$. It suffices to show that there exists such function $f_{\mathcal{A}}$ as well as a *time-then-graph* model $\{\mathbf{RNN}, \mathcal{A}'\}$ such that the following conditions hold:

For any graph G and its node $a,b \in V(G)$,

1. $\mathbb{H}^{V,(l)}_a = [\mathbf{x}_a, Enc(\{\!\{\mathbf{x}_{e_{aj}} | j \in \mathcal{N}(a)\}\!\})]$.

2.If there is at least one edge between a,b in history, $f_{\mathcal{A}}(\mathbb{H}^{V,(l)}_a, \mathbb{H}^{V,(l)}_b, \mathbb{H}^E_{ab}) = \mathbf{x}_{e_{ab}}$. Otherwise, $f_{\mathcal{A}}(\mathbb{H}^{V,(l)}_a, \mathbb{H}^{V,(l)}_b, \mathbb{H}^E_{ab}) = \mathbf{0}$

where $\mathbb{H}^{V,(l)}_a, \mathbb{H}^{V,(l)}_b$ are node features outputted by $\mathcal{A}'$, while $\mathbf{x}_a, \mathbf{x}_{e_{ab}}$ are node features outputted by $\mathcal{A}$ on transformed collpased graph. $Enc(\mathbf{X})$ is some injective encoding that stores all information of multiset $\mathbf{X}$. For a node feature $\mathbb{H}^{V,(l)}_a$ of above form, let $\mathbb{H}^{V,(l)}_{a,0} := \mathbf{x}_a, \mathbb{H}^{V,(l)}_{a,1} = Enc(\{\!\{\mathbf{x}_{e_{aj}} | j \in \mathcal{N}(a)\}\!\})$ denote two slices that store independent information in different positions.

For the base case $L = 0$. the node feature only depends on temporalized unary facts related to the corresponding node. Since by **RNN** we can use *most expressiveness representation* to capture all unary facts. A specific **RNN** already captures $\mathcal{A}$ when $L = 0$. Moreover, there is no added node e_{ab} that relates to any unary fact, so a constant function already satisfies the condition of $f_{\mathcal{A}}$ when $L = 0$. Therefore, our result holds for $L = 0$

Assume $L = k+1$, let $\widehat{\mathcal{A}}$ denote the model generated by the first k layers of $\mathcal{A}$. By induction, there is *time-then-graph* model $\widehat{\mathcal{A}}'$ and function $f_{\widehat{\mathcal{A}}'}$ that captures output of $\widehat{\mathcal{A}}'$ on transformed collapsed graph. We can append a layer to $\widehat{\mathcal{A}}'$ to build $\mathcal{A}'$ that simulates $\mathcal{A}$. Let $\{C^{(L)}, (A^{(L)}_j)^{T|P_2|}_{j=1}, A^{(L)}_{aux1}, A^{(L)}_{aux2}, R^{(L)}\}$ denote the building blocks of layer L of $\mathcal{A}$, and let u^*, g^*, r^* denote functions used in the layer that will be appended to $\widehat{\mathcal{A}}'$. They are defined below:

$$g^*(\{\!\{(\mathbb{H}^{V,(l-1)}_i, \mathbb{H}^{V,(l-1)}_j, \mathbb{H}^E_{ij} | j \in \mathcal{N}(i))\}\!\}) := A^{(L)}_{aux1}(\{\!\{f_{\widehat{\mathcal{A}}'}(\mathbb{H}^{V,(l-1)}_i, \mathbb{H}^{V,(l-1)}_j, \mathbb{H}^E_{ij}) | j \in \mathcal{N}(i)\}\!\}) \tag{48}$$

$$r^*(\{\!\{\mathbb{H}_j^{V,(l-1)}|j\in V(G)\}\!\}) = R^{(L)}\Big(\{\!\{\mathbb{H}_{j,0}^{V,(l-1)}|j\in V(G)\}\!\} \cup (\bigcup_{j\in V(G)} Dec(\mathbb{H}_{j,1}^{V,(l-1)}))\Big) \quad (49)$$

$$u^*(\mathbf{x}_g,\mathbf{x}_r) = C^{(L)}(\mathbf{x}_g,\mathbf{x}_r) \quad (50)$$

where $\mathbf{x}_g$,$\mathbf{x}_r$ are outputs of g^* and r^*. $Dec(\mathbf{X})$ is a decoder function that do inverse mapping of $Enc(\mathbf{X})$ mentioned above, so $Dec(\mathbb{H}_{j,1}^{V,(l-1)})$ is actually $\{\!\{\mathbf{x}_{e_{aj}}|j\in\mathcal{N}(a)\}\!\}$. Note that primal nodes in transformed graph only has type *aux1*- neighbors, so two inputs $\mathbf{x}_g$,$\mathbf{x}_r$, one for *aux1* aggregation output and one for global readout are already enough for computing the value. Comparing the three rules above with Equation (2), we can see that our new model $\mathcal{A}'$ perfectly captures $\mathcal{A}$.

We've captured $\mathcal{A}$, and the remaining work is to construct $f_{\mathcal{A}}$ defined above to complete inductive assumption. We can just choose a function that simulates message passing between pairs of added nodes e_{ab} and e_{ba} as well as message passing between e_{ab} and a, and that function satisfies the condition for $f_{\mathcal{A}}$. Formally speaking, $f_{\mathcal{A}}$ can be defined below:

$$f_{\mathcal{A}}(\mathbb{H}_i^{V,(l)},\mathbb{H}_j^{V,(l)},\mathbb{H}_{ij}^E) := \mathbf{Sim}_{\mathcal{A}_L}(\mathbb{H}_i^{V,(l-1)}, \mathbb{H}_g^{(l-1)},g_{ij},g_{ji}, \mathbb{H}_{ij}^E) \quad (51)$$

$$g_{ij} := f_{\widehat{\mathcal{A}}'}(\mathbb{H}_i^{V,(l-1)},\mathbb{H}_j^{V,(l-1)},\mathbb{H}_{ij}^E), \mathbb{H}_g^{(l-1)} := \{\!\{\mathbb{H}_i^{V,(l-1)}|i\in V(G)\}\!\} \quad (52)$$

Let's explain this equation, $\mathbf{Sim}_{\mathcal{A}_L}(a,g,s,b,e)$ is a local simulation function which simulates single-iteration message passing in the following scenario:

Suppose there is a graph H with three constants $V(H)=\{a,e_{ab},e_{ba}\}$. There is an *aux1* edge between a and e_{ab}, an *aux2* edge between e_{ab} and e_{ba}, and additional edges of different types between e_{ab} and e_{ba}. The description of additional edges can be founded in e. Initial node features of a,e_{ab},e_{ba} are set to a,s,b respectively. and the global readout output is g. Finally, run L-th layer of $\mathcal{A}$ on H, and $\mathbf{Sim}_{\mathcal{A}_L}$ is node feature of e_{ab} outputted by $\mathcal{A}_L$.

Note that if we use appropriate injective encoding or just use concatenation technic, $\mathbb{H}_g^{(l-1)}$, $\mathbb{H}_i^{V,(l-1)}$,$\mathbb{H}_j^{V,(l-1)}$ can be accessed from $\mathbb{H}_i^{V,(l)}$,$\mathbb{H}_i^{V,(l)}$. Therefore the above definition for $f_{\mathcal{A}}$ is well-defined. Moreover, in the above explanation we can see that $f_{\mathcal{A}}(\mathbb{H}_i^{V,(l-1)},\mathbb{H}_j^{V,(l-1)},\mathbb{H}_{ij}^E)$ is exactly node feature of e_{ij} outputted by $\mathcal{A}$ on the transformed collapsed graph, so our proof finishes.

A.9 Proof of Theorem 14

Theorem 14. *R^2-TGNN $\circ F^T = R^2$-TGNN $\circ F \circ H$ on any universal graph class $\mathcal{G}_u$.*

First, we recall the definition for R^2-TGNN as in Equation (53):

$$\mathbf{x}_v^t = \mathcal{A}_t\Big(G_t,v,\mathbf{y}^t\Big) \qquad \text{where} \qquad \mathbf{y}_v^t = [I_{G_t}(v) : \mathbf{x}_v^{t-1}], \forall v\in V(G_t) \quad (53)$$

We say a R^2-TGNN is *homogeneous* if $\mathcal{A}_1,\dots,\mathcal{A}_T$ share the same parameters. In particular, we first prove Lemma 28, namely, *homogeneous* R^2-TGNN and R^2-TGNN (where paramters in $\mathcal{A}_1,\dots,\mathcal{A}_T$ may differ) have the same expressiveness.

Lemma 28. *homogenous R^2-TGNN = R^2-TGNN*

Proof. The forward direction *homogeneous* R^2-TGNN$\subseteq$ R^2-TGNN trivially holds. It suffices to prove the backward direction.

Let $\mathcal{A} : \{\mathcal{A}_t\}_{t=1}^T$ denote a R^2-TGNN. Without loss of generality, we can assume all models in each timestamps have the same layer number L. Then for each $1\le t\le T$, we can assume all $\mathcal{A}_t$ can be represented by $\{C_t^{(l)},(A_{t,j}^{(l)})_{j=1}^{|P_2|},R_t^{(l)}\}_{l=1}^L$. Futhormore, without loss of generality, we can assume all output dimensions for $A_{t,j}^{(l)}$,$R_t^{(l)}$ and $C_t^{(l)}$ are d. As for input dimension, all of these functions also have input dimension d for $2\le l\le L$. Specially, by updating rules of R^2-TGNN Equation (53), in the initialization stage of each timestamp we have to concat a feature with length $|P_1|$ to output of the former timestamp, so the input dimension for $A_{t,j}^{(1)}$,$R_t^{(1)}$,$C_t^{(1)}$ is $d+|P_1|$.

We can construct an equivalent *homogeneous* R^2-TGNN with L layers represented by $\{C^{*,(l)}, (A_j^{*,(l)})_{j=1}^{|P_2|}, R^{*,(l)}\}_{l=1}^{L}$. For $2 \leq l \leq L$, $C^{*,(l)} A_j^{*,(l)}, R^{*,(l)}$ use output and input feature dimension $d' = Td$. Similar to the discussion about feature dimension above, since we need to concat the unary predicates information before each timestamp, for layer $l = 1$, $C^{*,(1)}, A_j^{*,(1)}, R^{*,(1)}$ have input dimension $d' + |P_1|$ and output dimension d'. For dimension alignment, $\mathbf{x}_v^0$ used in Equation (53) is defined as zero-vector with length d'.

Next let's define some symbols for notation simplicity. For a feature vector $\mathbf{x}$, let $\mathbf{x}[i,j]$ denotes the slice of $\mathbf{x}$ in dimension $[i,j]$. By the discussion above, in the following construction process we will only need feature $\mathbf{x}$ with dimension d' or $d' + |P_1|$. When $\mathbf{x}$ has dimension d', $\mathbf{x}^{(i)}$ denotes $\mathbf{x}[(i-1)d+1, id]$, otherwise it denotes $\mathbf{x}[|P_1| + (i-1)d + 1, |P_1| + id]$. Let $[\mathbf{x}_1......\mathbf{x}_T]$ or $[\mathbf{x}_t]_{t=1}^T$ denotes the concatenation of a sequence of feature $\mathbf{x}_1......\mathbf{x}_T$, and $[\mathbf{x}]^n$ denote concatenation of n copies of $\mathbf{x}$, $\mathbf{0}^n$ denotes zero vectors of length n. Furthermore. Let $\mathbf{X}$ denotes a multiset of $\mathbf{x}$. Follows the updating rules defined in Equation (2), for all $1 \leq j \leq |P_2|, 1 \leq l \leq L$, $A_j^{*,(l)}, R^{*,(l)}$ should get input of form $\mathbf{X}$, and the combination function $C^{*,(l)}$ should get input of form $(\mathbf{x}_0, (\mathbf{x}_j)_{j=1}^{|P_2|}, \mathbf{x}_g)$, where $\mathbf{x}_0$ is from the node itself, $(\mathbf{x}_j)_{j=1}^{|P_2|}$ are from aggregation functions $(A_j^{*,(l)})_{j=1}^{|P_2|}$ and $\mathbf{x}_g$ is from the global readout $R^{*,(l)}$. The dimension of $\mathbf{x}$ or $\mathbf{X}$ should match the input dimension of corresponding function. For all $1 \leq l \leq L$, parameters in layer l for the new model are defined below

$$l = 1 : C^{*,(l)}(\mathbf{x}_0, (\mathbf{x}_j)_{j=1}^{|P_2|}, \mathbf{x}_g) := [C_t^{(l)}([\mathbf{x}_0[1, |P_1|], \mathbf{x}_0^{(t-1)}], (\mathbf{x}_j^{(t)})_{j=1}^{|P_2|}, \mathbf{x}_g^{(t)})]_{t=1}^T \tag{54}$$

$$2 \leq l \leq L : C^{*,(l)}(\mathbf{x}_0, (\mathbf{x}_j)_{j=1}^{|P_2|}, \mathbf{x}_g) := [C_t^{(l)}(\mathbf{x}_0^{(t)}, (\mathbf{x}_j^{(t)})_{j=1}^{|P_2|}, \mathbf{x}_g^{(t)})]_{t=1}^T \tag{55}$$

$$\forall j \in [K], l = 1 : A_j^{*,(l)}(\mathbf{X}) = [A_{t,j}^{(l)}(\{\!\{[\mathbf{x}[1, |P_1|], \mathbf{x}^{(t-1)}] | \mathbf{x} \in \mathbf{X}\}\!\})]_{t=1}^T \tag{56}$$

$$l = 1 : R^{*,(l)}(\mathbf{X}) = [R_t^{(l)}(\{\!\{[\mathbf{x}[1, |P_1|], \mathbf{x}^{(t-1)}] | \mathbf{x} \in \mathbf{X}\}\!\})]_{t=1}^T \tag{57}$$

$$\forall j \in [K], 2 \leq l \leq L : A_j^{*,(l)}(\mathbf{X}) = [A_{t,j}^{(l)}(\{\!\{\mathbf{x}^{(t)} | \mathbf{x} \in \mathbf{X}\}\!\})]_{t=1}^T \tag{58}$$

$$2 \leq l \leq L : R^{*,(l)}(\mathbf{X}) = [R_t^{(l)}(\{\!\{\mathbf{x}^{(t)} | \mathbf{x} \in \mathbf{X}\}\!\})]_{t=1}^T \tag{59}$$

The core trick is to use T disjoint slices $\mathbf{x}^{(1)}......\mathbf{x}^{(T)}$ to simulate T different models $\mathcal{A}_1......\mathcal{A}_T$ at the same time, Since these slices are isolated from each other, a proper construction above can be found. The only speciality is that in layer $l = 1$, we have to incorporate the unary predicate information $\mathbf{x}[1, |P_1|]$ into each slice. By the construction above, we can see that for any node v, $\mathbf{x}_v^{(T)}$ is exactly the its feature outputted by $\mathcal{A}$. Therefore, we finally construct an *homogeneous* R^2-TGNN equivalent with $\mathcal{A}$. □

Now, we start to prove Theorem 14.

Theorem 14. *R^2-TGNNs $\circ F^T$ = R^2-GNNs $\circ F \circ H$ on any universal graph class $\mathcal{G}_u$.*

Proof. Since R^2-TGNN $\circ F^T$ only uses a part of predicates of $P' = F(H(P))$ in each timestamp, the forward direction R^2-TGNN $\circ F^T \subseteq$ R^2-GNN $\circ F \circ H$ trivially holds.

For any R^2-GNN $\mathcal{A}$ under P', we want to construct an R^2-TGNN $\mathcal{A}'$ under $F^T(P)$ such that for any temporal knowledge graph G, $\mathcal{A}'$ outputs the same feature vectors as $\mathcal{A}$ on $F^T(G)$. We can assume $\mathcal{A}$ is represented as $(C^{(l)}, (A_j^{(l)})_{j=1}^K, A_{aux1}^{(l)}, A_{aux2}^{(l)}, R^{(l)})_{l=1}^L$, where $K = T|P_2|$.

First, by setting feature dimension to be $d' = T|P| + 3$. We can construct an R^2-TGNN $\mathcal{A}'$ whose output feature stores all facts in $F(H(G))$ for any graph G. Formally speaking, $\mathcal{A}'$ should satisfy the following condition:

For any primal node a, its feature outputted by $\mathcal{A}' \circ F^T$ should store all unary facts of form $A_i(a), A_i \in T|P_1|$ or $primal(a)$ on $F(H(G))$. For any non-primal node e_{ab}, its feature outputted by $\mathcal{A}' \circ F^T$ should store all binary facts of form $r_i(a,b), r_i \in T|P_2|$ or $r_{aux1}(a,b), r_{aux2}(a,b)$ where b is another node on $F(H(G))$.

The $\mathcal{A}'$ is easy to construct since we have enough dimension size to store different predicates independently, and these facts are completely encoded into the initial features of corresponding timestamp. Let $(\mathcal{A}'_1......\mathcal{A}'_T)$ denote $\mathcal{A}'$.

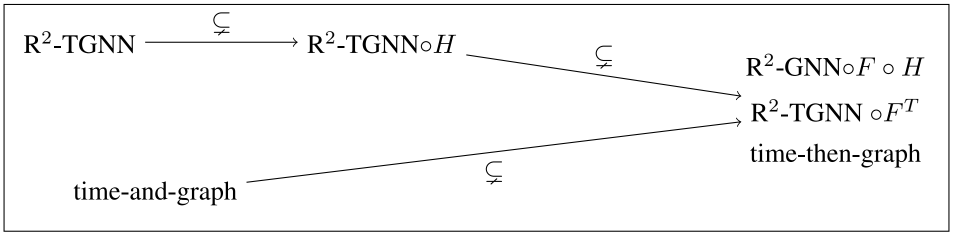

Figure 6: Hierarchic expressiveness.

Next, in order to simulate $\mathcal{A}$, we need to append some layers to $\mathcal{A}'_T$. Let L denote the layer number of $\mathcal{A}$, we need to append L layers represented as $(C^{*,(l)},(A_j^{*,(l)})_{j=1}^{|P_2|},A_{aux1}^{*,(l)},A_{aux2}^{*,(l)},R^{*,(l)})_{l=1}^{L}$

Since we have enough information encoded in features, we can start to simulate $\mathcal{A}$. Since neighbor distribution of primal nodes don't change between $F^T(G)_T$ and $F(H(G))$, it's easy to simulate all messages passed to primal nodes as destinations by $A_{aux1}^{*,(l)}$. For messages passed to non-primal node e_{ab} as destination, it can be divided into messages from a and messages from e_{ba}. The first class of messages is easy to simulate since the $aux1$ edge between e_{ab} and a is the same on $F^T(G)_T$ and $F(H(G))$.

For the second class of messages, since edges of type r_i,$1 \leq i \leq T|P_2|$ may be lost in $F^T(G)_T$, we have to simulate these messages only by the unchanged edge of type **aux2**. It can be realized by following construction:

$$1 \leq l \leq L, A_{aux2}^{*,(l)}(\mathbf{X}) = [[A_j^{\prime,(l)}(\mathbf{X}))]_{j=1}^{K}, A_{aux2}^{(l)}(\mathbf{X})] \tag{60}$$

where $K = T|P_2|$, $A_j^{\prime,(l)}(\mathbf{X}) := A_j^{(l)}(\mathbf{X})$ if and only if e_{ba} has neighbor r_j on $F(H(G))$, otherwise $A_j^{\prime(l)}(\mathbf{X}) := \mathbf{0}$. Note that $\mathbf{X}$ is exactly the feature of e_{ba}, and we can access the information about its r_j neighbors from feature since $\mathcal{A}'$ has stored information about these facts.

In conclusion, we've simulated all messages between neighbors. Furthermore, since node sets on $F^T(G)_T$ and $F(H(G))$ are the same, global readout $R^{(l)}$ is also easy to simulate by $R^{*,(l)}$. Finally, using the original combination function $C^{(l)}$, we can construct an R^2-TGNN on F^T equivalent to $\mathcal{A}$ on $F(H(G))$ for any temporal knowledge graph G.

□

A.10 An expressiveness hierarchy

In the main body of this paper, we give expressiveness comparison among R^2-TGNN $\circ F^T$ *time-and-graph* and *time-then-graph*. However, we don't calibrate expressiveness of some weaker frameworks such as R^2-TGNN, R^2-TGNN $\circ H$. On the other hand, in experiment part **??** we introduce a logical classifier φ_3 but don't explain why it can't be captured by R^2-TGNN. So it's necessary to calibrate expressiveness of these weaker framework and build a hierarchy here.

Theorem 29. *If time range $T > 1$ R^2-TGNN $\subsetneq$ R^2-GNN $\circ H$.*

Proof. Since in each timestamp t, R^2-TGNN only uses a part of predicates in temporalized predicate set $P' = H(P)$, R^2-TGNN $\subseteq$ R^2-GNN $\circ H$ trivially holds. To show R^2-TGNN is strictly weaker than R^2-GNN $\circ H$. Consider the following classifier:

Let time range $T = 2$, and let r be a binary predicate in P_2. Note that there are two different predicates r^1,r^2 in $P' = H(P)$. Consider the following temporal graph G with 5 nodes $\{1,2,3,4,5\}$. its two snapshots G_1,G_2 are as follows:

$G_1 = \{r(1,2), r(4,5)\}$

$G_2 = \{r(2,3)\}$.

It follows that after transformation H, the static version of G is:

$H(G) = \{r_1(1,2), r_1(4,5), r_2(2,3)\}$.

datasets	φ_1	φ_2	φ_3	φ_4
Avg # Nodes	477	477	477	477
Time_range	2	2	2	10
# Unary predicate	2	2	2	3
# Binary predicate(non-temporalized)	1	1	1	3
Avg # Degree (in single timestamp)	3	3	3	5
Avg # positive percentage	50.7	52	25.3	73.3

Table 4: statistical information for synthetic datasets.

datasets	AIFB	MUTAG	Brain-10
# Nodes	8285	23644	5000
Time_Range	\	\	12
# Relation types	45	23	20
# Edges	29043	74227	1761414
# Classes	4	2	10
# Train Nodes	140	272	4500
# Test Nodes	36	68	500

Table 5: statistical information for Real datasets.

Consider the logical classifier $\exists y\Big(r_1(x,y) \wedge (\exists x r_2(x,y))\Big)$ under P'.It can be captured by some R^2-GNN under P'. Therefore, R^2-GNN $\circ H$ can distinguish nodes 1,4.

However, any R^2-TGNN based on updating rules in Equation (53) can't distinguish these two nodes, so R^2-TGNN is strictly weaker than R^2-GNN $\circ H$. □

Based on Theorem 29, we can consider logical classifier φ_3 defined in **??**: $\varphi_{\mathbf{3}} \coloneqq \exists^{\geq 2} y(p_1^1(x,y) \wedge p_1^2(x,y))$. Note that this classifier is just renaming version of Figure 1. Therefore φ_3 can't be captured by R^2-GNN $\circ H$, not to say weaker framework R^2-GNN by Theorem 29.

Finally, we give a strict expressiveness hierearchy as follows:

Corollary 29.1. *If time range $T > 1$ R^2-GNN $\subsetneq$ R^2-GNN $\circ H$ $\subsetneq$ R^2-TGNN $\circ F \circ H$ = R^2-TGNN $\circ F^T$*

Proof. It's a conclusion based on Theorem 29, Figure 3 and Theorem 14. □

Combining Corollary 29.1 and Theorem 13, our final expressiveness hierarchy is as Figure 3.

B Experiment supplementary

B.1 Synthetic dataset generation

For each synthetic datasets, we generate 7000 graphs as tranining set and 500 graphs as test set. Each graph has $50 - 1000$ nodes. In graph generation, we fix the expected edge density δ. In order to generate a graph with n nodes, we pick δn pairs of distinct nodes uniformly randomly. For each selected node pair a,b, each timestamp t and each binary relation type r, we add $r^t(a,b)$ and $r^t(a,b)$ into the graph with independent probability $\frac{1}{2}$.

B.2 statistical information for datasets

We list the information for synthetic dataset in Table 4 and real-world dataset in Table 5. Note that synthetic datasets contains many graphs, but real-world datasets only contains a single graph. Therefore, for real-world dataset, we have two disjoint node set as train split and test split for training and testing respectively. In training, the model can see the subgraph induced by train split and unlabelled nodes, in testing, the model can see the whole graph but only evaluate the performance on test split.

hyper-parameter	range
learning rate	0.01
combination	mean/max/add
aggregation/readout	mean/max/add
layer	1,2,3
hidden dimension	10,64,100

Table 6: Hyper-parameters.

$\mathcal{FOC}_2$ classifier	φ_1			φ_2			φ_3			φ_4		
Aggregation	sum	max	mean	sum	max	mean	sum	max	mean	sum	max	mean
Temporal Graphs Setting												
R-TGNN	100	60.7	65.4	61.0	51.3	52.4	93.7	82.3	84.4	83.5	60.0	61.3
R^2-TGNN	100	63.5	66.8	93.1	57.7	60.2	94.5	83.3	85.9	85.0	62.3	66.2
R^2-GNNs $\circ F^T$	**100**	67.2	68.1	**99.0**	57.6	62.2	**100**	88.8	89.2	**98.1**	73.4	77.5
Aggregated Static Graphs Setting												
R-GNNs $\circ H$	100	61.2	69.9	62.3	51.3	55.5	94.7	80.5	83.2	80.2	60.1	60.4
R^2-GNNs $\circ H$	100	62.7	66.8	92.4	56.3	58.5	95.5	84.2	85.2	81.0	58.3	64.5
R^2-GNNs $\circ F \circ H$	**100**	70.2	70.8	**98.8**	60.6	60.2	**100**	85.6	86.5 7	**95.5**	70.3	79.7

Table 7: Test set node classification accuracies (%) on synthetic temporal multi-relational graphs datasets and their aggregated static multi-relational graphs datasets. The best results are highlighted for two different settings.

B.3 hyper-parameters

For all experiments, we did grid search according to Table 6.

Realworld dataset	AIFB			MUTAG			Brain-10		
Aggregation	sum	max	mean	sum	max	mean	sum	max	mean
Temporal Graphs Setting									
R-TGNN	\	\	\	\	\	\	**85.0**	82.3	82.8
R^2-TGNN	\	\	\	\	\	\	**94.8**	82.3	91.0
R^2-GNNs $\circ F^T$	\	\	\	\	\	\	**94.0**	83.5	92.5
Aggregated Static Graphs Setting									
R-GNNs	**91.7**	73.8	82.5	**76.5**	63.3	73.2	\	\	\
R^2-GNNs	**91.7**	73.8	82.5	**85.3**	62.1	79.5	\	\	\
R^2-GNNs $\circ F$	**97.2**	75.0	89.2	**88.2**	65.5	82.1	\	\	\

Table 8: Test set node classification accuracies (%) on realworld temporal multi-relational graphs datasets and multi-edge datasets. The best results are highlighted for two different settings.