Learning Rule-Induced Subgraph Representations for Inductive Relation Prediction

Tianyu Liu, Qitan Lv, Jie Wang, Shuling Yang, Hanzhu Chen

University of Science and Technology of China

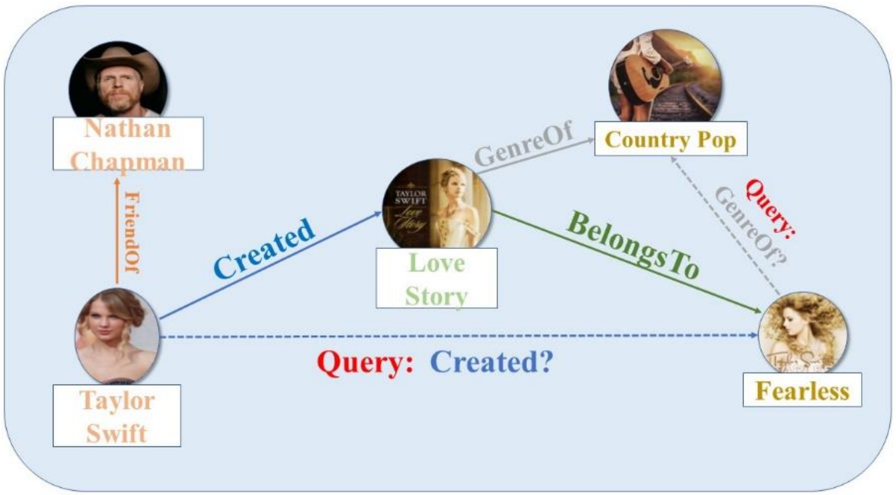
Overlook

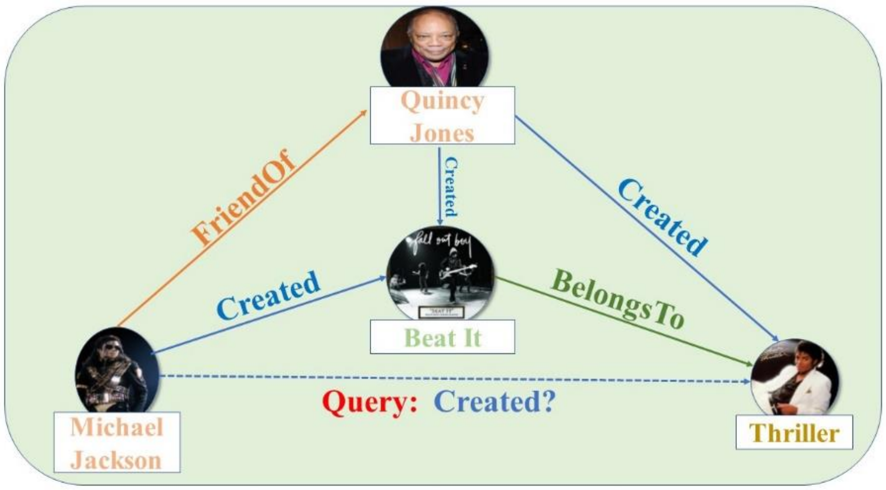
- Introduction and Motivation
- Methods
- Theoretical Analysis and Findings
- Experiments

Introduction and Motivation

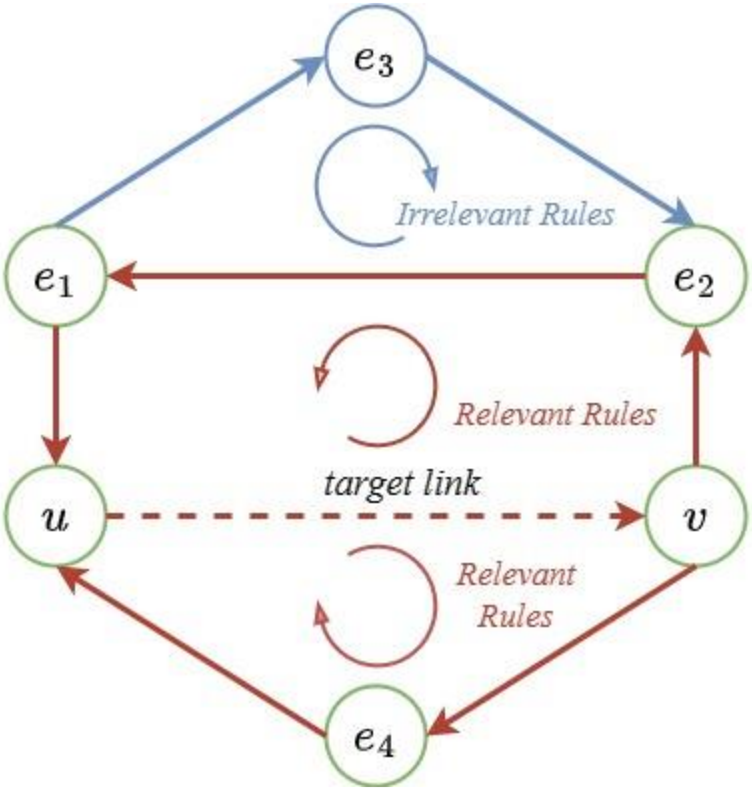
- Objective: Predicting missing relations in evolving knowledge graphs.

- Goal: Training on one graph and test on another graph.





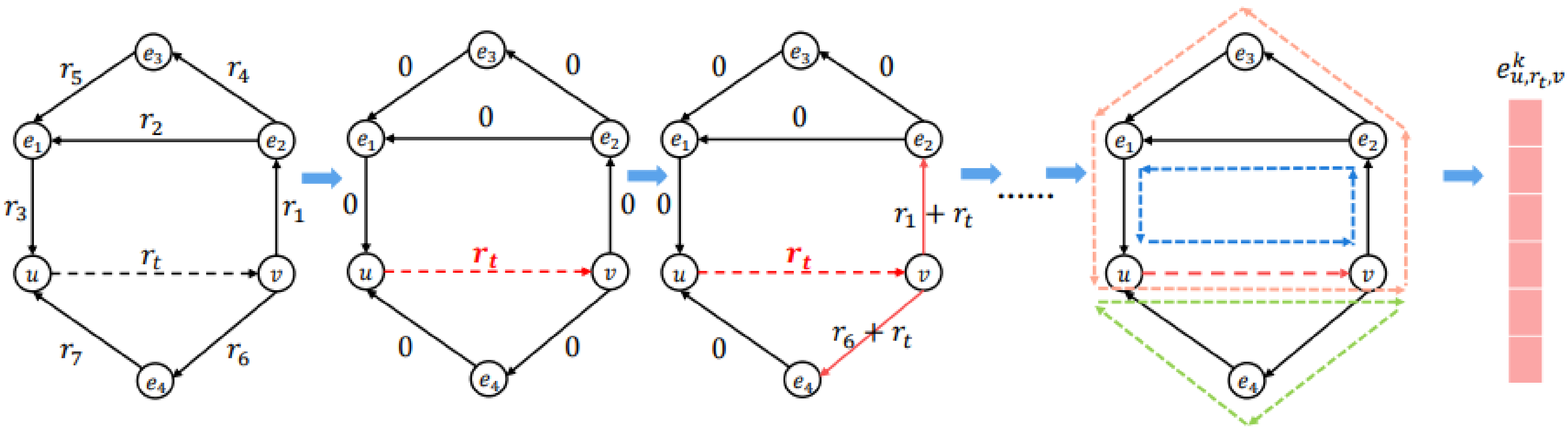
Introduction and Motivation



- ***Relevant rules*** are cycles that pass the target link, while ***irrelevant rules*** are on the contrary.
- Existing methods cannot **differentiate** relevant rules from irrelevant ones, resulting in suboptimal representations with irrelevant information.

Method

- We propose a novel single-source edge-wise GNN REST, which **encodes** relevant rules and **eliminates** irrelevant rules within the subgraph.



1. Extracted local subgraph for the target link (u, r_t, v).

2. Single-source initialization for all edges within the subgraph.

3. Edge-wise message passing to update edge representations. Finally, we output the representation of the target link $e^k_{u,r_t,v}$ to calculate its plausibility.

Method

Single-source Initialization:

$$\mathbf{e}^0_{x,y,z} = \underset{(u,r,v)}{\mathbb{1}}(x,y,z) \odot \mathbf{r}_y = \begin{cases} \mathbf{r}_y, & \text{if } (x,y,z) = (u,r_t,v) \\ \mathbf{0}, & \text{if } (x,y,z) \neq (u,r_t,v) \end{cases} \qquad (1)$$

$$\mathbf{h}^0_v = \mathbf{0} \qquad for\ \forall v \in \mathcal{E},$$

Edge-wise Message Passing:

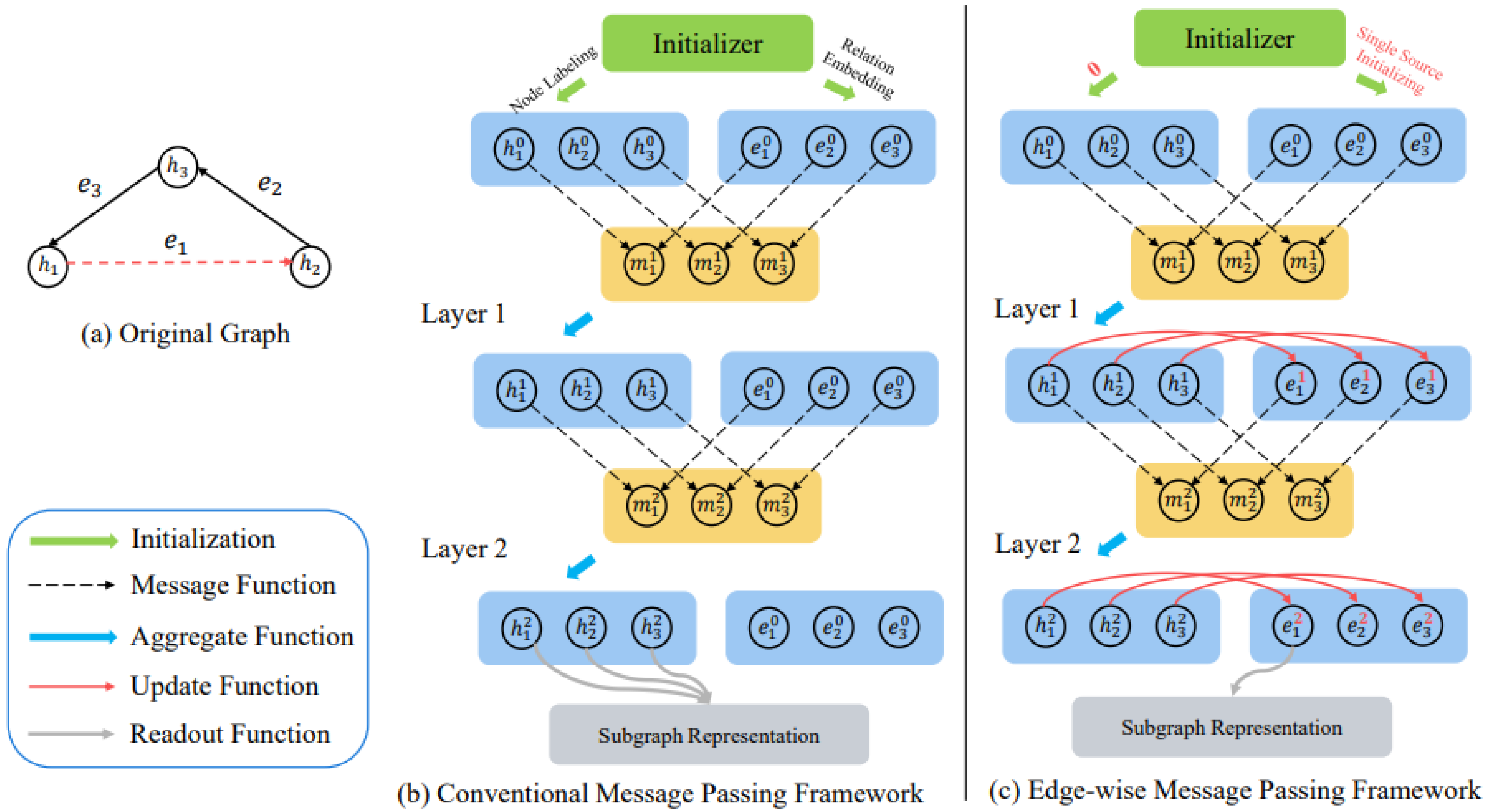
$$\mathbf{m}^k_{x,y,z} = \textsc{Message}(\mathbf{h}^{k-1}_x, \mathbf{e}^{k-1}_{x,y,z}, \mathbf{r}_y) = (\mathbf{h}^{k-1}_x \otimes^1 \mathbf{r}_y) \uplus (\mathbf{e}^{k-1}_{x,y,z} \otimes^2 \mathbf{r}_y) \qquad (2)$$

$$\mathbf{h}^k_z = \underset{(x,y,z)\in\mathcal{T}}{\textsc{Aggragate}}(\mathbf{m}^k_{x,y,z}) = \bigoplus_{(x,y,z)\in\mathcal{T}} \mathbf{m}^k_{x,y,z} \qquad (3)$$

$$\mathbf{e}^k_{x,y,z} = \textsc{Update}(\mathbf{h}^k_x, \mathbf{e}^{k-1}_{x,y,z}) = \mathbf{h}^k_x \diamond \mathbf{e}^{k-1}_{x,y,z} \qquad (4)$$

Theoretical Analysis and Findings

- Our REST is basically different with GraIL.



Theoretical Analysis and Findings

- Our REST is theoretically effective.

Definition 1 (Rule-induced subgraph representation.) *Given a subgraph $\mathcal{SG}_{u,r_t,v}$, its rule-induced subgraph representation is defined as follows:*

$$\mathcal{S}_{u,r_t,v} = \bigoplus_{i=1}^{k} \underbrace{\bigoplus_{(u,r_t,v)} \bigoplus_{(v,y_0,x_0)} \cdots \bigoplus_{(x_{i-3},y_{i-2},u)}}_{i} \alpha_{i1}\mathbf{r}_{r_t} \otimes \alpha_{i2}\mathbf{r}_{y_0} \otimes \ldots \otimes \alpha_{ii}\mathbf{r}_{y_{i-2}} \tag{11}$$

where i denotes the length of the cycle, $\mathbf{r}_{y_i}$ is the representation of relation y_i, (x_i, y_i, x_{i-1}) is an existing triple in $\mathcal{SG}_{u,r_t,v}$. $\{(u, r_t, v), (v, y_0, x_0), \ldots, (x_{i-3}, y_{i-2}, u)\}$ is a cycle at length i.

Theorem 2 *Single-source edge-wise GNN can learn rule-induced subgraph representation if $\uplus = \oplus, \oplus = \oplus, \diamond = \oplus, \otimes^1 = \otimes, \otimes^2 = \otimes$, where $\oplus$ and $\otimes$ are binary operators that satisfy $0 \oplus a = a, 0 \otimes a = 0$. i.e., there exists nonzero $\alpha_{i,j}$ such that*

$$\mathbf{e}^{k}_{u,r_t,v} = \bigoplus_{i=1}^{k} \underbrace{\bigoplus_{(u,r_t,v)} \bigoplus_{(v,y_0,x_0)} \cdots \bigoplus_{(x_{i-3},y_{i-2},u)}}_{i} \alpha_{i1}\mathbf{r}_{r_t} \otimes \alpha_{i2}\mathbf{r}_{y_0} \otimes \ldots \otimes \alpha_{ii}\mathbf{r}_{y_{i-2}} \tag{13}$$

Experiments

- REST is effective:

		WN18RR				FB15k-237				NELL-995			
		v1	v2	v3	v4	v1	v2	v3	v4	v1	v2	v3	v4
Rule-Based	Neural LP	74.37	68.93	46.18	67.13	52.92	58.94	52.90	55.88	40.78	78.73	82.71	80.58
	DRUM	74.37	68.93	46.18	67.13	52.92	58.73	52.90	55.88	19.42	78.55	82.71	80.58
	RuleN	80.85	78.23	53.39	71.59	49.76	77.82	87.69	85.60	53.50	81.75	77.26	61.35
Subgraph-Based	GraIL	82.45	78.68	58.43	73.41	64.15	81.80	82.83	89.29	59.50	93.25	91.41	73.19
	CoMPILE	83.60	79.82	60.69	75.49	67.64	82.98	84.67	87.44	58.38	93.87	92.77	75.19
	TACT	84.04	81.63	67.97	76.56	65.76	83.56	85.20	88.69	79.80	88.91	94.02	73.78
	SNRI	87.23	83.10	67.31	83.32	71.79	86.50	89.59	89.39	-	-	-	-
	ConGLR	85.64	92.93	70.74	92.90	68.29	85.98	88.61	89.31	81.07	94.92	94.36	81.61
	REST(ours)	**96.28**	**94.56**	**79.50**	**94.19**	**75.12**	**91.21**	**93.06**	**96.06**	**88.00**	**94.96**	**96.79**	**92.61**

- REST outperforms existing rule-based and subgraph-based methods by a large margin

Experiments

- REST is efficient:

		WN18RR				FB15k-237				NELL-995			
		v1	v2	v3	v4	v1	v2	v3	v4	v1	v2	v3	v4
Enclosing Subgraph	GraIL	121.77	537.42	1127.14	194.98	949.48	2933.04	8423.59	15089.74	136.55	1197.24	6112.77	1303.97
	REST	54.01	251.97	617.16	91.76	111.34	338.24	868.79	1,626.77	61.19	213.45	688.14	219.33
	Efficiency	**2.25**×	**2.13**×	**1.83**×	**2.12**×	**8.53**×	**8.67**×	**9.70**×	**9.28**×	**2.23**×	**5.61**×	**8.88**×	**5.95**×
Unclosing Subgraph	GraIL	127.69	517.94	1194.18	199.00	1287.35	4166.63	11499.32	21738.29	167.06	1611.97	8044.53	1542.82
	REST	56.27	260.20	631.71	95.23	123.36	386.55	985.81	1890.54	64.72	245.41	858.23	248.00
	Efficiency	**2.27**×	**1.99**×	**1.89**×	**2.09**×	**10.44**×	**10.78**×	**11.66**×	**11.50**×	**2.58**×	**6.57**×	**9.37**×	**6.22**×

- REST significantly accelerates the subgraph preprocessing time by up to 11.66× (6.02× on average).

Experiments

- Case study

Table 3: Some relations and their top-3 relevant relations. The relations are taken from WN18RR.

Rule Head	Rule Body	Scores
_hypernym	_similar_to -> _hypernym^{-1}	0.89
	_also_see-> _hypernym^{-1}	0.84
	_hypernym^{-1}->_also_see->_verb_group	0.67
_derivationally_related_form	_derivationally_related_form->_also_see	0.82
	_has_part− > _also_see	0.76
	_derivationally_related_form-> _also_see-> _has_part	0.66
_member_meronym	_has_part-> _synset_domain_topic_of	0.77
	has_part-> _synset_domain_topic_of-> _derivationally_related_form	0.76
	_has_part-> _derivationally_related_form	0.71
_synset_domain_topic_of	_has_part-> _also_see	0.95
	_synset_domain_topic_of^{-1}-> _similar_to	0.91
	_has_part-> _synset_domain_topic_of^{-1}	0.81

Thank You!