Inductive Relation Prediction by Subgraph Reasoning

Published as a conference paper at ICML 2020

Komal K. Teru, Etienne Denis, William L. Hamilton

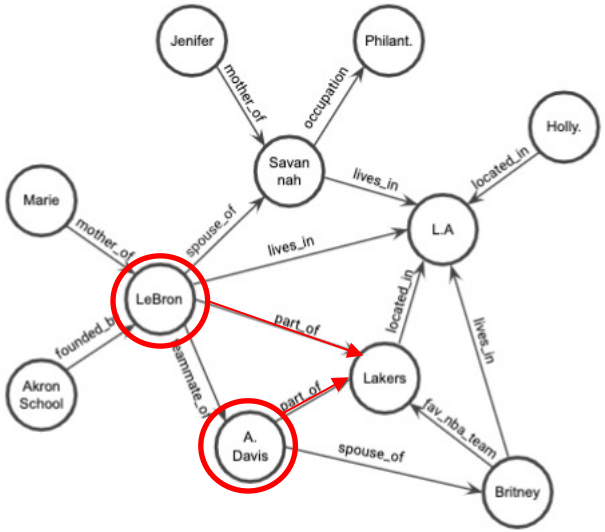
2024-10-11

HoonUi Lee

Index

Previous work

- Embedding-based model



LeBron and A.Davis belong to the Lakers

=> Use to predict they are teammates

❖ Local connectivity pattern

- Embedding-based methods place similar entities in close
- Allows to reflect the connection of node with another neighbor

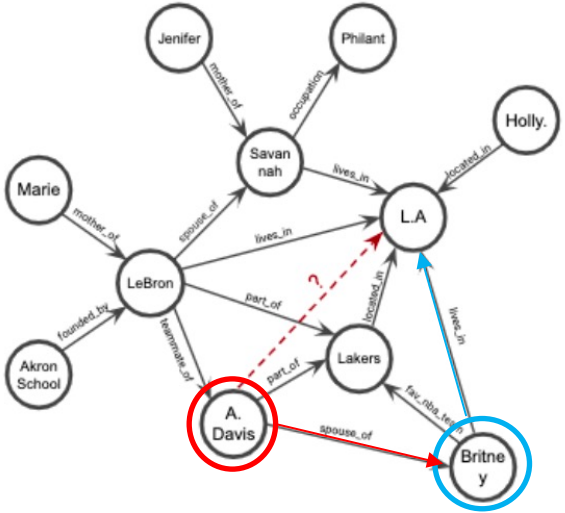
❖ Homophily

- Because similar entities are placed in close,
  Homophily between entities can be well reflected
  and effective in clustering

➢ **e.g) Anyone who closely associated with the Lakers would live in L.A**

Previous work

- Limitation of Embedding-based model



Britney is A. Davis' wife

If Britney lives in L.A.,

A. Davis will live in L.A

❖ BUT in capturing Relational Semantics..

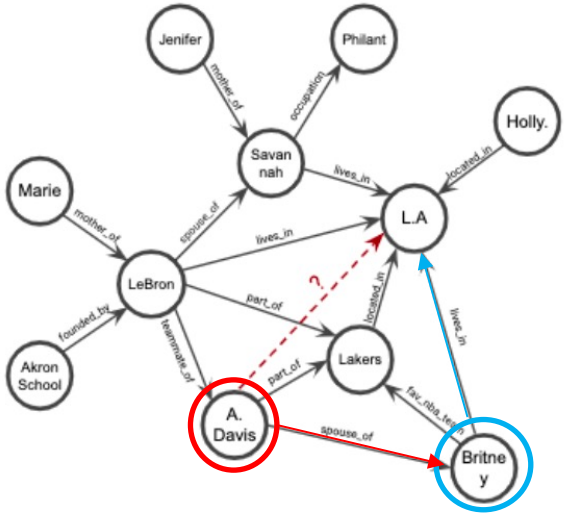
- Is it possible to extract **Logical rule** from given Knowledge Graph?

$$\exists Y.(X, \texttt{spouse\_of}, Y) \wedge (Y, \texttt{lives\_in}, Z) \rightarrow (X, \texttt{lives\_in}, Z)$$

- Logical rules capture **entity-independent** relational meaning
- There are limitations to the **Entity-specific** embedding-based model

Previous work

- Limitation of Embedding-based model



(A.Davis, spouse_of, Britney)

(Britney, lives_in, L.A.)

- Prediction about **Entities** that are **already given in training data**

➢ **Transductive**

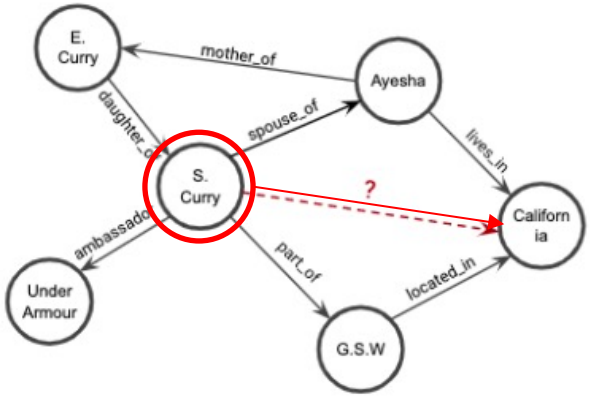
- Learning the connection between specific entities by 'lives_in', 'spouse_of'
- Can't learning the logical semantic between 'lives_in, 'spouse_of'

➢ **Entity-specific**

Previous work

- Limitation of Embedding-based model

Unseen graph



c. Inductive inference

❖ **Testing about data that was not in training data**

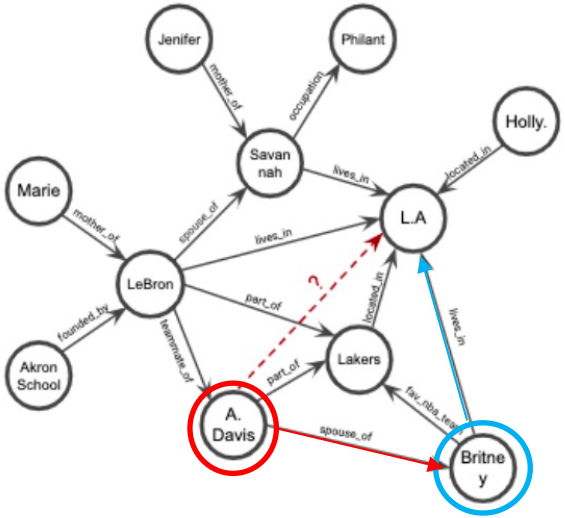
- Gives **unseen graph**
  and test to predict relation between S. Curry and California

➢ These entities were not given in training dataset

➢ Embedding-based model can't predict the relation 'lives_in'

Previous work

- logical rule-based model



Britney is A. Davis' wife

If Britney lives in L.A.,

A. Davis will live in L.A

(X, spouse_of, Y), (Y, lives_in, Z)

→ (X, lives_in, Z)

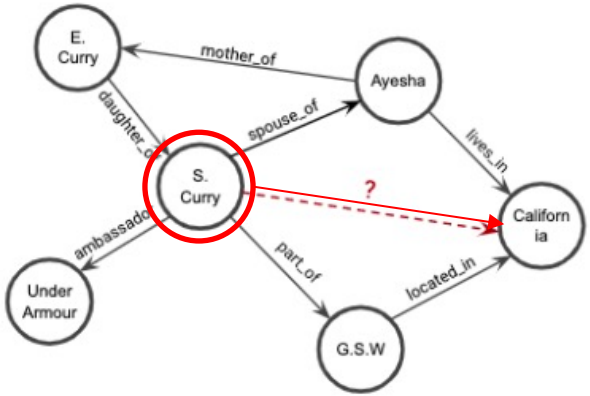
- Learn **latent rules in Knowledge Graph** with existing relationship combinations
- Learn the various combinations that a 'live_in' relationship itself can have

➢ **Entity-independent**

Previous work

- logical rule-based model

Unseen graph



c. Inductive inference

❖ **Testing about data that was not in training data**

- Gives **unseen graph**
  and test to predict relation between S. Curry and California

➢ If model learned following logical rules
(X, spouse_of, Y) ∧ **(Y, lives_in, Z)** → **(X, lives_in, Z)** in previously seen graph, then, model can infer that Curry is likely to lives in California

➢ While this unseen graph and Entities are not in training data

Previous work

- Inductive ability

❖ Limitations of the existing logical rule derivation method

- **Long time to derive** and apply logical rules on **large graphs**
- **Lack of expressiveness** for complex relationships compared to embedding methods that can express entities and relations in **high-dimensional vector spaces**

GraIL

❖ Predict relation between two nodes by using Subgraph

- Analyzing the structure of subgraph enclosing two specific nodes
- Use **only structure information** without attribute of nodes when using GNN
- Predicting the likelihood of a possible relation r_t between head node u and in triplet (u, r_t, v)

❖ Three sub-task

- i) **Extracting the enclosing subgraph** around the target nodes
- ii) **labeling** the nodes in the extracted subgraph
- iii) **Scoring** the labeled subgraph **using a GNN**

GraIL

❖ **Three sub-task**

- i) **Extracting the enclosing subgraph** around the target nodes
- ii) **labeling** the nodes in the extracted subgraph
- iii) **Scoring** the labeled subgraph **using a GNN**

❖ **Limiting the scope of learning within a subgraph**

- Applicate well on large graphs

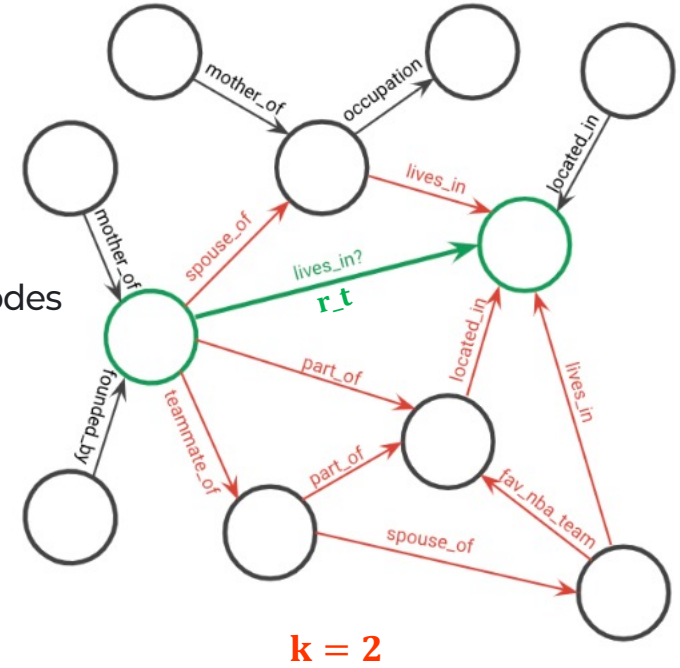
❖ **Learn the structural patterns between nodes within a subgraph using GNN**

- Consider learned rules between specific nodes + other connections within a subgraph
- Increase in expression than existing rule-based models

GraIL

❖ Extracting Subgraph

- Assume that **local graph neighborhood** will contain **logical evidence** needed to deduce the relation between target nodes
- **Remove unnecessary nodes** from the intersection between each set of neighborhood nodes of two target nodes
- Compute the enclosing graph by $N_k(u) \cap N_k(v)$ and then prune nodes that are isolated or at a distance greater than k from target nodes

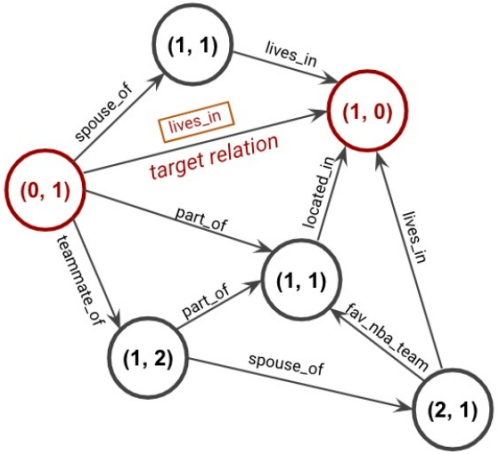


1. Sample the enclosing sub-graph around the link to be predicted (target link).

GraIL

❖ Node labeling

- Create node feature matrix required for GNN through labeling
- Node i in the subgraph, is labeled with the tuple $(\mathbf{d(i,u)}, \mathbf{d(i,v)})$ that denotes the shortest distance between nodes i and nodes u, v
- Convert this shortest distance by **one - hot encoding** $[one-hot(d(i,u)) \oplus one-hot(d(i,v))]$ and calculate final vector



2. Label the nodes w.r.t the target nodes to identify their structural role. Uniquely labels target nodes to mark them for the model.

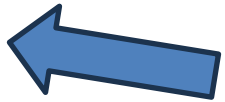
- ✓ In this case, $d(i,\mathrm{u})$ and $d(i,\mathrm{v})$ are the shortest distance between node i and u, and the shortest distance between node i and v
- ✓ Node u and v are labeled $(0,1)$ and $(1,0)$, respectively

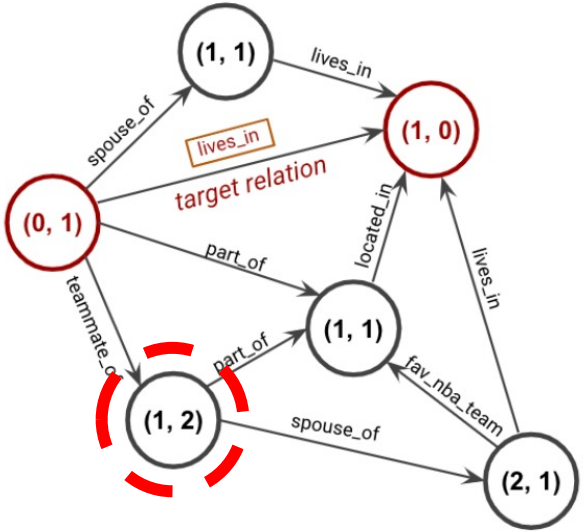
GraIL

- Node Labeling

- $\left[one - hot\left(d(i,u)\right) \oplus one\ - hot\left(d(i,v)\right)\right]$
- $\left[[0,1,0] \oplus [0,0,1]\right]$
- $[0,1,0,0,0,1]$

➢ **Node labeling vector**





2. Label the nodes w.r.t the target nodes to identify their structural role. Uniquely labels target nodes to mark them for the model.

GraIL

- GNN Scoring

❖ Aggregate each node representation with its neighborhoods'

$$\mathbf{a}_t^k = \text{AGGREGATE}^k\left(\left\{\mathbf{h}_s^{k-1} : s \in \mathcal{N}(t)\right\}, \mathbf{h}_t^{k-1}\right)$$
$$\mathbf{h}_t^k = \text{COMBINE}^k\left(\mathbf{h}_t^{k-1}, \mathbf{a}_t^k\right),$$

- Combine the neighbors node representation of the previous step
  with its own node representation to create the updated representation of the next step k

✓ $N(t)$ means the neighbors of node t
✓ h_t^k denotes the latent representation of node t in the k-th layer
✓ In this case, h_i^0 (each node's initial representation) initialized to the labeled matrix about structural information

GraIL

- GNN Scoring

- ✓ W_r^k is the transformation matrix used to propagate messages
- ✓ Adopt the basis sharing mechanism for W_r^k -> preventing overfitting
- ✓ Edge dropout in aggregating information from the neighborhood

❖ AGGREGATE function

- Calculate the attention weight for each neighboring node s on node t

$$\mathbf{a}_t^k = \sum_{r=1}^{R} \sum_{s \in \mathcal{N}_r(t)} \alpha_{rr_t st}^k \mathbf{W}_r^k \mathbf{h}_s^{k-1}$$

$$\alpha_{rr_t st}^k = \sigma\left(\mathbf{A}_2^k \mathbf{s} + \mathbf{b}_2^k\right)$$

- Combine node representation of t and s, learned attention embeddings of relations $e_r^a, e_{r_t}^a$

$$\mathbf{s} = \text{ReLU}\left(\mathbf{A}_1^k [\mathbf{h}_s^{k-1} \oplus \mathbf{h}_t^{k-1} \oplus \mathbf{e}_r^a \oplus \mathbf{e}_{r_t}^a] + \mathbf{b}_1^k\right)$$

GraIL

- GNN Scoring

✓ Learning weight matrix W_{self}^{k} for itself in the course

❖ COMBINE function

$$\mathbf{h}_t^k = \mathrm{ReLU}\left(\mathbf{W}_{self}^k \mathbf{h}_t^{k-1} + \mathbf{a}_t^k\right)$$

- Combine the representation of node t in the layer from the previous step, the weight matrix for itself and the message received from neighborhood nodes

- Update representation by gathering about neighbors while reflecting itself

GraIL

- GNN Scoring

❖ Subgraph Representation

$$\mathbf{h}^{L}_{\mathcal{G}_{(u,v,r_t)}} = \frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\mathbf{h}^{L}_{i}$$

- Generating the representation vector of subgraph
  as the average of the final representation vectors of each node through L layers

- Integrating and using **all node information in a subgraph**, not just reflecting target nodes u, v,
  captures important structural interactions that may be missed when only target nodes are considered

GraIL

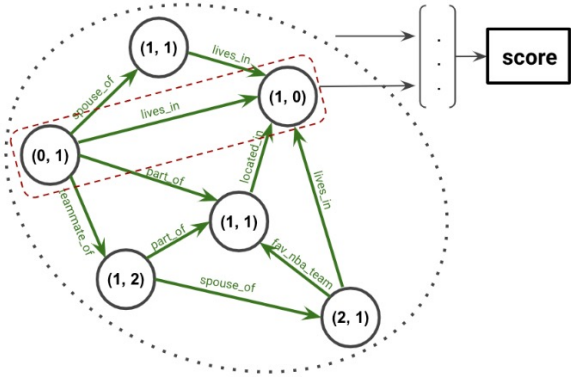
- GNN Scoring

✓ With JK-connection, get information across a wide range of layers without decreased performance even with more layers

❖ Triplet Scoring

$$\text{score}(u, r_t, v) = \mathbf{W}^T [\mathbf{h}^L_{\mathcal{G}_{(u,v,r_t)}} \oplus \mathbf{h}^L_u \oplus \mathbf{h}^L_v \oplus \mathbf{e}_{r_t}]$$

- Combine the representation vector of subgraph, the target node u, the v
  and the embedding vector of target relation
- Calculate score through linear layer W^T



3. Pass messages across the (sub-)graph to predict a score indicating how strongly the structure around the target link supports its logical plausibility.

- Using JK-connection mechanism

$$\text{score}(u, r_t, v) = \mathbf{W}^T \bigoplus_{i=1}^{L} [h^i_{\mathcal{G}_{(u,v,r_t)}} \oplus h^i_u \oplus h^i_v \oplus e_{r_t}]$$

➢ Use the representation obtained from the intermittent layer as well as the last layer

GraIL

❖ Is GraIL learning with logical rules?

$$\alpha^k_{rr_tst} = \sigma\left(\mathbf{A}^k_2 \mathbf{s} + \mathbf{b}^k_2\right)$$

$$\mathbf{s} = \mathrm{ReLU}\left(\mathbf{A}^k_1 [\mathbf{h}^{k-1}_s \oplus \mathbf{h}^{k-1}_t \oplus \mathbf{e}^a_r \oplus \mathbf{e}^a_{r_t}] + \mathbf{b}^k_1\right)$$

- Node representation with structural information of nodes,
  target relation's attention embedding, and current node-to-node relationship's attention embedding
  are considered together in the learning process

➢ Learn which path (relations between nodes) is best for a specific target relationship

GraIL

❖ Is GraIL learning with logical rules?

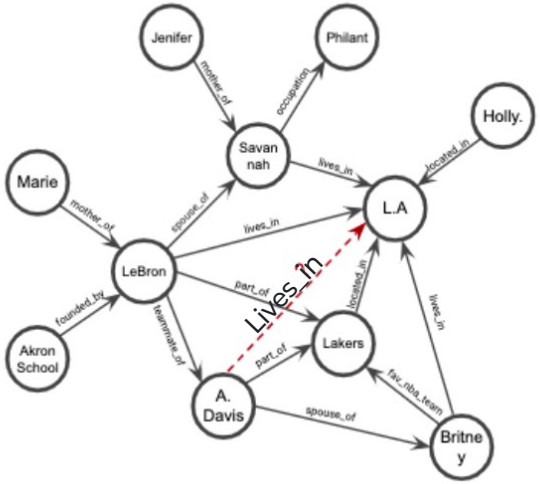
$$\alpha^k_{rr_tst} = \sigma\left(\mathbf{A}^k_2\mathbf{s} + \mathbf{b}^k_2\right)$$

$$\mathbf{s} = \mathrm{ReLU}\left(\mathbf{A}^k_1[\mathbf{h}^{k-1}_s \oplus \mathbf{h}^{k-1}_t \oplus \mathbf{e}^a_r \oplus \mathbf{e}^a_{r_t}] + \mathbf{b}^k_1\right)$$

- Node representation with structural information of nodes,
  target relation's attention embedding, and current node-to-node relationship's attention embedding
  are considered together in the learning process

➢ Learn which path (relations between nodes) is best for a specific target relationship

GraIL



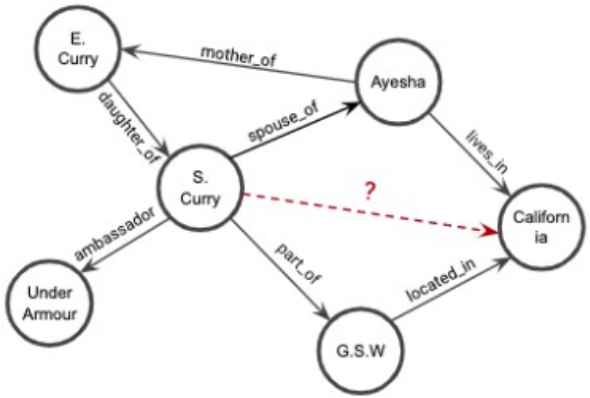
$$\alpha^k_{rr_tst} = \sigma\left(\mathbf{A}^k_2\mathbf{s} + \mathbf{b}^k_2\right)$$

$$\mathbf{s} = \mathrm{ReLU}\left(\mathbf{A}^k_1[\mathbf{h}^{k-1}_s \oplus \mathbf{h}^{k-1}_t \oplus \mathbf{e}^a_r \oplus \mathbf{e}^a_{r_t}] + \mathbf{b}^k_1\right)$$

- [(0, 1), (1, 1)] linked by 'spouse_of' relation
- [(1, 1), (1, 0)] linked by 'lives_in' relation
- Learn to scoring high on this path
- lives_in (X, Z) ← spouse_of (X, Y) ∧ lives_in (Y, Z)

- [(0, 1), (1, 1)] linked by 'part_of' relation
- [(1, 1), (1, 0)] linked by 'located_in' relation
- Learn to scoring high on this path
- lives_in (X, Z) ← part_of (X, Y) ∧ located_in (Y, Z)

GraIL



c. Inductive inference

$$\alpha^k_{rr_tst} = \sigma\left(\mathbf{A}^k_2\mathbf{s} + \mathbf{b}^k_2\right)$$

$$\mathbf{s} = \text{ReLU}\left(\mathbf{A}^k_1[\mathbf{h}^{k-1}_s \oplus \mathbf{h}^{k-1}_t \oplus \mathbf{e}^a_r \oplus \mathbf{e}^a_{r_t}] + \mathbf{b}^k_1\right)$$

❖ **Suppose the following rules are being learned by the model**

lives_in (X, Z) ← spouse_of (X, Y) ∧ lives_in (Y, Z)
lives_in (X, Z) ← part_of (X, Y) ∧ located_in(Y, Z)
daughter_of (X, Z) ← spouse_of (X, Y) ∧ mother_of (Y, Z)

- Scoring (S. Curry, **daughter_of**, California)
  - ➢ [(0, 1), (1, 1)] linked by 'spouse_of' relation
  - ➢ [(1, 1), (1, 0)] linked by '**lives_in**' relation

$e^a_{daughter\_of}$ and $e^a_{lives\_in}$ reflects less connectivity in learned rule

with information of nodes structure h^{k-1}_{Ayesha}, $h^{k-1}_{California}$

- ➢ Get low score

GraIL

- Training Regime

✓ Create a negative triplet
by randomly replacing one of the triplet's head and tail

❖ Loss function

$$\mathcal{L} = \sum_{i=1}^{|\mathcal{E}|} \max(0, \text{score}(n_i) - \text{score}(p_i) + \gamma)$$

- Loss function based on the **score difference** between positive and negative triplets
- Higher scores for negative triplets result in losses
- The γ is margin hyperparameter, the minimum score difference between the two triplets

GraIL

- Inference Complexity

- ✓ V: number of nodes in subgraph
- ✓ ϵ: number of edges in subgraph
- ✓ R: number of relations in subgraph
- ✓ d: dimension of node / relation embeddings
- ✓ k: number of GNN layer

❖ Complexity

$$O(\log(\mathcal{V})\mathcal{E} + \mathcal{R}dk)$$

Dijkstra — Subgraph learning by GNN

- Evaluate the shortest path from the target node to all other nodes within the extracted subgraph
- The inference cost is largely determined by the size of the extracted subgraphs
- Actual execution time is dominated by running the **Dijkstra algorithm**

Experiment

❖ Inductive relation prediction

- Compares rule induction performance to existing models in inductive setting

❖ Transductive relation prediction

- Compares rule induction performance to existing embedding-based models in transductive setting
- Compares performance while do Late / Early fusion ensemble GraIL with exisiting embedding-based models

❖ Ablation Study

- Test importance of three subtasks performance contribution

Experiment

- Inductive relation prediction

❖ Build new fully-inductive benchmark dataset

- Existing datasets were developed by transductive setting
  - ➢ Entities included in the training data are also included in the test data

- Extracted the subgraph train and ind-test to make the dataset
- Only share relations with between train and ind-test and configure entities as disjoint
- Choose 10% edge from ind-test-graph during the actual test

Table 13. Statistics of inductive benchmark datasets

		WN18RR			FB15k-237			NELL-995		
		#relations	#nodes	#links	#relations	#nodes	#links	#relations	#nodes	#links
v1	train	9	2746	6678	183	2000	5226	14	10915	5540
	ind-test	9	922	1991	146	1500	2404	14	225	1034
v2	train	10	6954	18968	203	3000	12085	88	2564	10109
	ind-test	10	2923	4863	176	2000	5092	79	4937	5521
v3	train	11	12078	32150	218	4000	22394	142	4647	20117
	ind-test	11	5084	7470	187	3000	9137	122	4921	9668
v4	train	9	3861	9842	222	5000	33916	77	2092	9289
	ind-test	9	7208	15157	204	3500	14554	61	3294	8520

Experiment

- Inductive relation prediction

- **Compare with NeuralLP and DRUM**

  Compare between End-to-End differentiable methods in inductive setting

- **Compare with RuleN**

  the current state-of-the-art rule mining model

Table 1. Inductive results (AUC-PR)

	WN18RR				FB15k-237				NELL-995			
	v1	v2	v3	v4	v1	v2	v3	v4	v1	v2	v3	v4
Neural-LP	86.02	83.78	62.90	82.06	69.64	76.55	73.95	75.74	64.66	83.61	87.58	85.69
DRUM	86.02	84.05	63.20	82.06	69.71	76.44	74.03	76.20	59.86	83.99	87.71	85.94
RuleN	90.26	89.01	76.46	85.75	75.24	88.70	91.24	91.79	84.99	88.40	87.20	80.52
GraIL	**94.32**	**94.18**	**85.80**	**92.72**	**84.69**	**90.57**	**91.68**	**94.46**	**86.05**	**92.62**	**93.34**	**87.50**

Table 2. Inductive results (Hits@10)

	WN18RR				FB15k-237				NELL-995			
	v1	v2	v3	v4	v1	v2	v3	v4	v1	v2	v3	v4
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
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

- ✓ AUC-PR (Precision-Recall): Create the same number of negative samples as the actual triplet present in the test set to score the triplet
- ✓ Hits@10: Score after creating 50 negative samples in one positive sample, If a positive sample being ranked in the top 10, then hit
- ✓ Average of 5-runs

Experiment

- Transductive relation prediction

❖ **How GraIL performs in the transductive setting**

❖ **The utility of ensembling GraIL with existing embedding-based approaches**

❖ **Late ensemble**

- Calculate each score that two models predict true and false for a particular triplet
- Form the feature vector for each test point ex) TransE: 0.7, GraIL: 0.8 ➔ [0.7, 0.8]
- Put the created feature vector into the linear classifier to classify whether it is true or false.
  (linear classifier is trained to score the true triplets higher that the negative triplets)

➢ Train linear classifier using the validation set

Experiment

- Transductive relation prediction

Table 3. Late fusion ensemble results on WN18RR (AUC-PR)

	TransE	DistMult	ComplEx	RotatE	GraIL
T	93.73	93.12	92.45	93.70	94.30
D		93.08	93.12	93.16	95.04
C			92.45	92.46	94.78
R				93.55	94.28
G					90.91

Table 4. Late fusion ensemble results on NELL-995 (AUC-PR)

	TransE	DistMult	ComplEx	RotatE	GraIL
T	98.73	98.77	98.83	98.71	98.87
D		97.73	97.86	98.60	98.79
C			97.66	98.66	98.85
R				98.54	98.75
G					97.79

Table 5. Late fusion ensemble results on FB15k-237 (AUC-PR)

	TransE	DistMult	ComplEx	RotatE	GraIL
T	98.54	98.41	98.45	98.55	97.95
D		97.63	97.87	98.40	97.45
C			97.99	98.43	97.72
R				98.53	98.04
G					92.06

: without late fusion ensemble

: Ensemble with GraIL

Experiment

- Transductive relation prediction

Table 6. Early fusion ensemble with TrasnE results (AUC-PR)

	WN18RR	FB15k-237	NELL-995
GraIL	90.91	92.06	97.79
GraIL++	96.20	93.91	98.11

❖ **Early fusion ensemble (GraIL++, in Transductive setting)**

- Concatenate the embedding of nodes obtained by the TransE model after node labeling

Experiment

- Ablation study

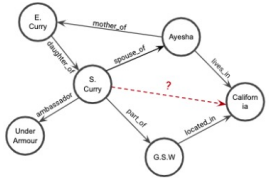
Table 7. Ablation study of the proposed framework (AUC-PR)

	FB (v3)	NELL (v3)
GraIL	**91.68**	**93.34**
GraIL w/o enclosing subgraph	84.25	85.89
GraIL w/o node labeling scheme	82.07	84.46
GraIL w/o attention in GNN	90.27	87.30

❖ **Score for each situation w/o three subtasks (AUC-PR, in inductive relation prediction)**

- In case of w/o enclosing subgraph, extract nodes from target node w/ k-hop
- In case of w/o node labeling scheme, fix all node embedding by (1, 1)
- In case of w/o attention in GNN, calculate node representation w/o attention mechanism

Conclusion



c. Inductive inference

Previous work

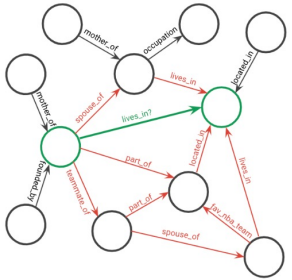
Existing **embedding-based models** are difficult to learn **based on logical rules.**

Existing **logical rule-based model** took longer and **lacked expressive.**

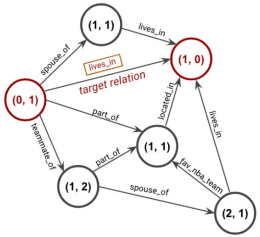
GraIL

In **GraIL, learning about subgraphs** overcomes the problem of graph size,

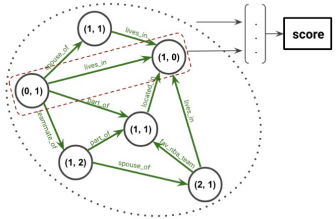
and **learn various structural patterns** between nodes within a subgraph **using GNN.**



1. Sample the enclosing sub-graph around the link to be predicted (target link).



2. Label the nodes w.r.t the target nodes to identify their structural role. Uniquely labels target nodes to mark them for the model.



3. Pass messages across the (sub-)graph to predict a score indicating how strongly the structure around the target link supports its logical plausibility.

Experiments

good performance in inductive setting.

performance improvement through ensemble with existing embedding-based model.

tested about GraIL's three subtask's importance.