AdaProp: Learning Adaptive Propagation for Graph Neural Network based Knowledge Graph Reasoning

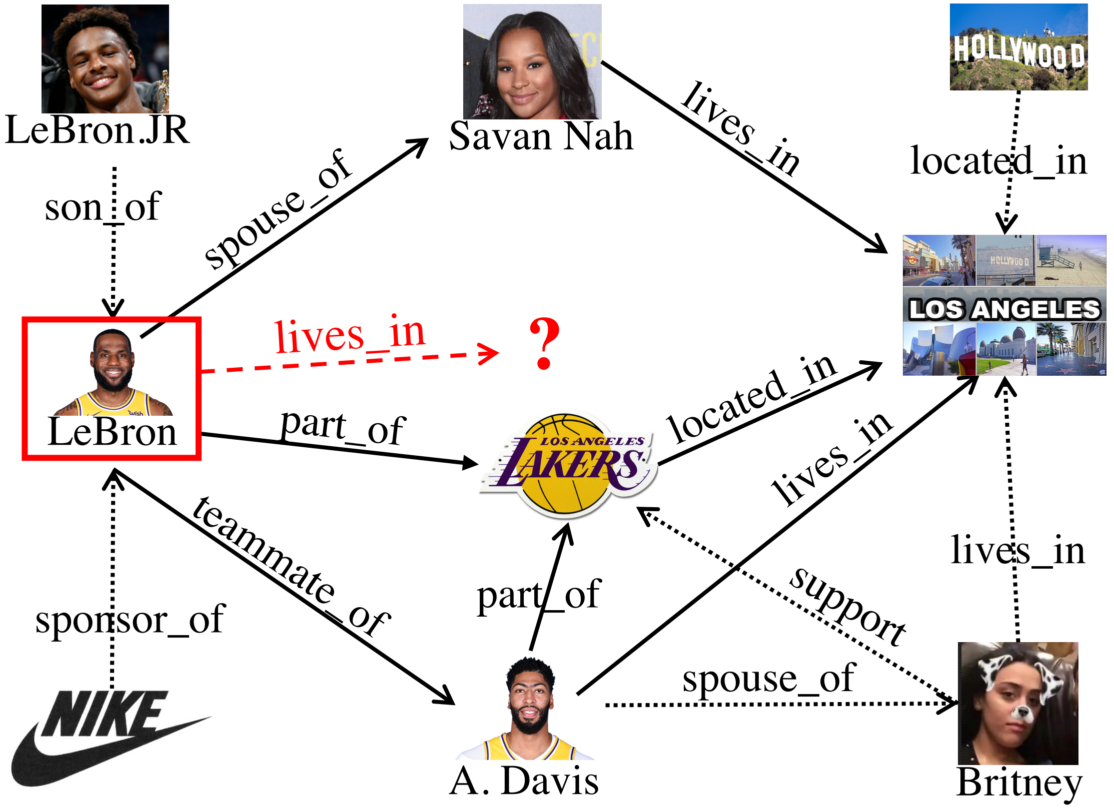
Yongqi Zhang*, Zhanke Zhou*, Quanming Yao, Xiaowen Chu, Bo Han

* zhangyongqi@4paradigm.com & cszkzhou@comp.hkbu.edu.hk in equal contrution

Outline

- Background
- Method
- Experiments
- Summary

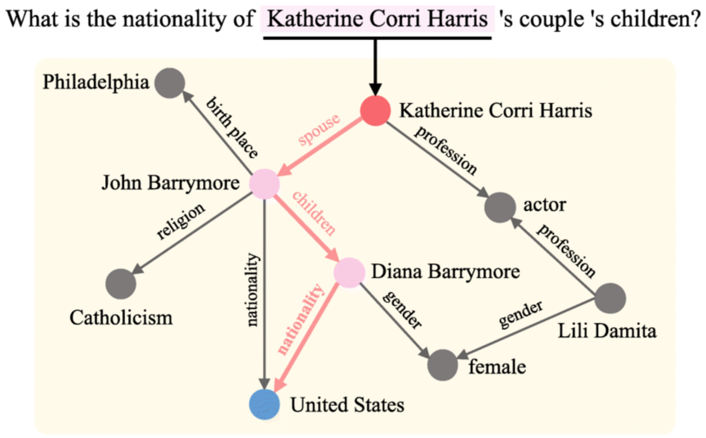
Knowledge Graph Reasoning



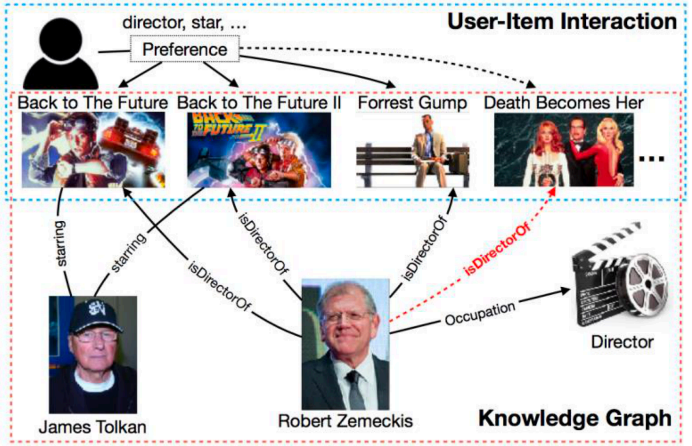
KG reasoning with query: (LeBron, lives_in, ?)

Applications

KGQA:

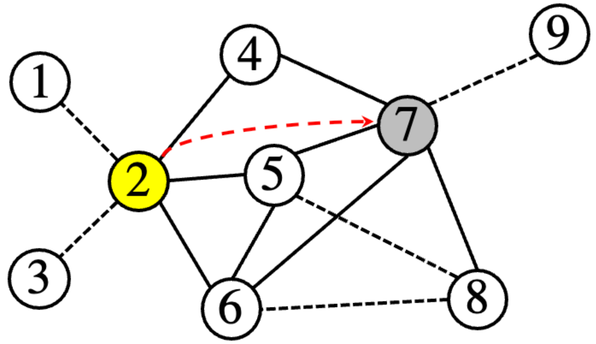


Recommendation:



Data structure for KG Reasoning

- Three classes of existing works
  - triple-based
    - (*head, relation, tail*)
  - path-based
    - $e_q \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \xrightarrow{r_l} e_l \xrightarrow{r_{l+1}} \cdots$
  - graph-based



The focus!

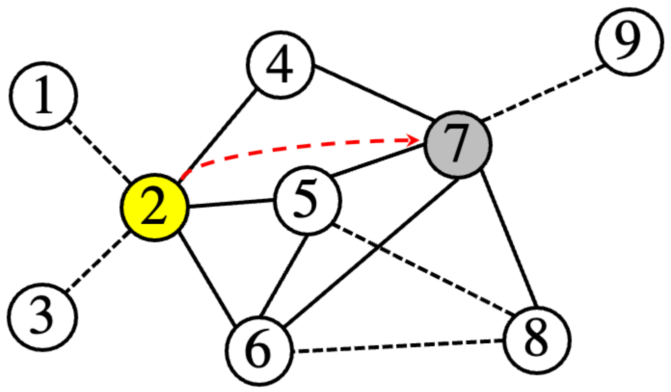
$$p(e_q, r_q, e_a) \rightarrow [0,1]$$

Encoding the corresponding data structure, and mapping the representation into the probability of query triplets (e_q, r_q, e_a).

GNN for KG Reasoning

- Graph-based method for KG reasoning
  - **propagate** the message with the graph structure
  - **update** entity representation at each propagation step

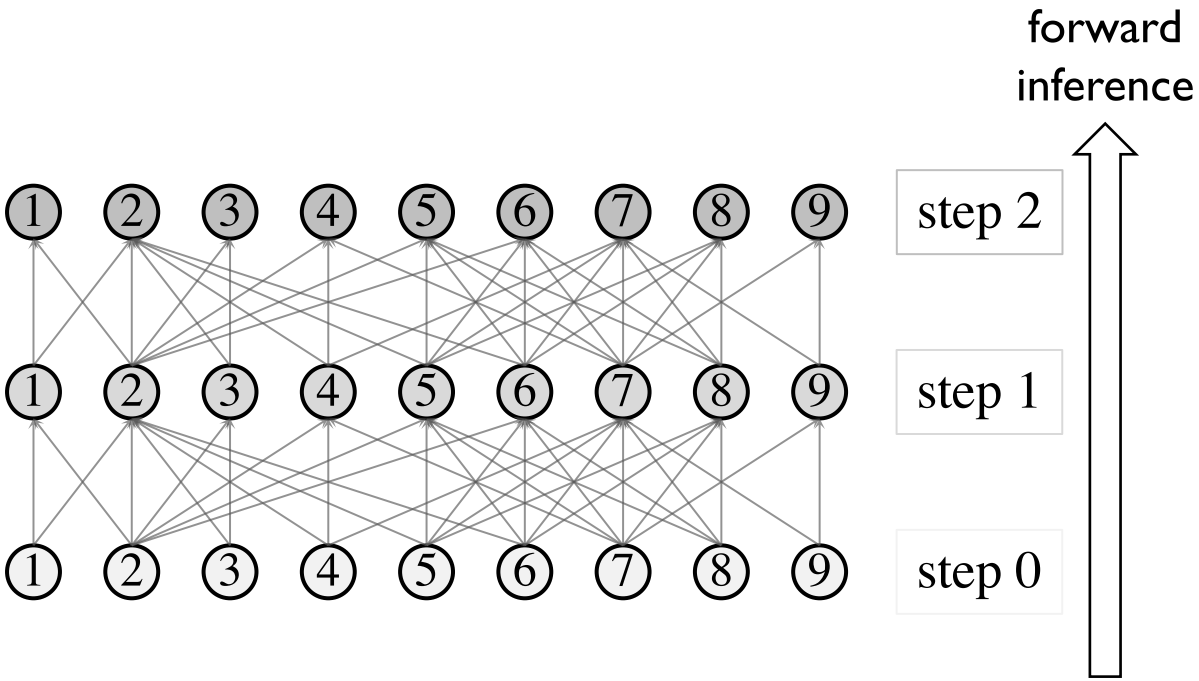
Graph / Subgraph



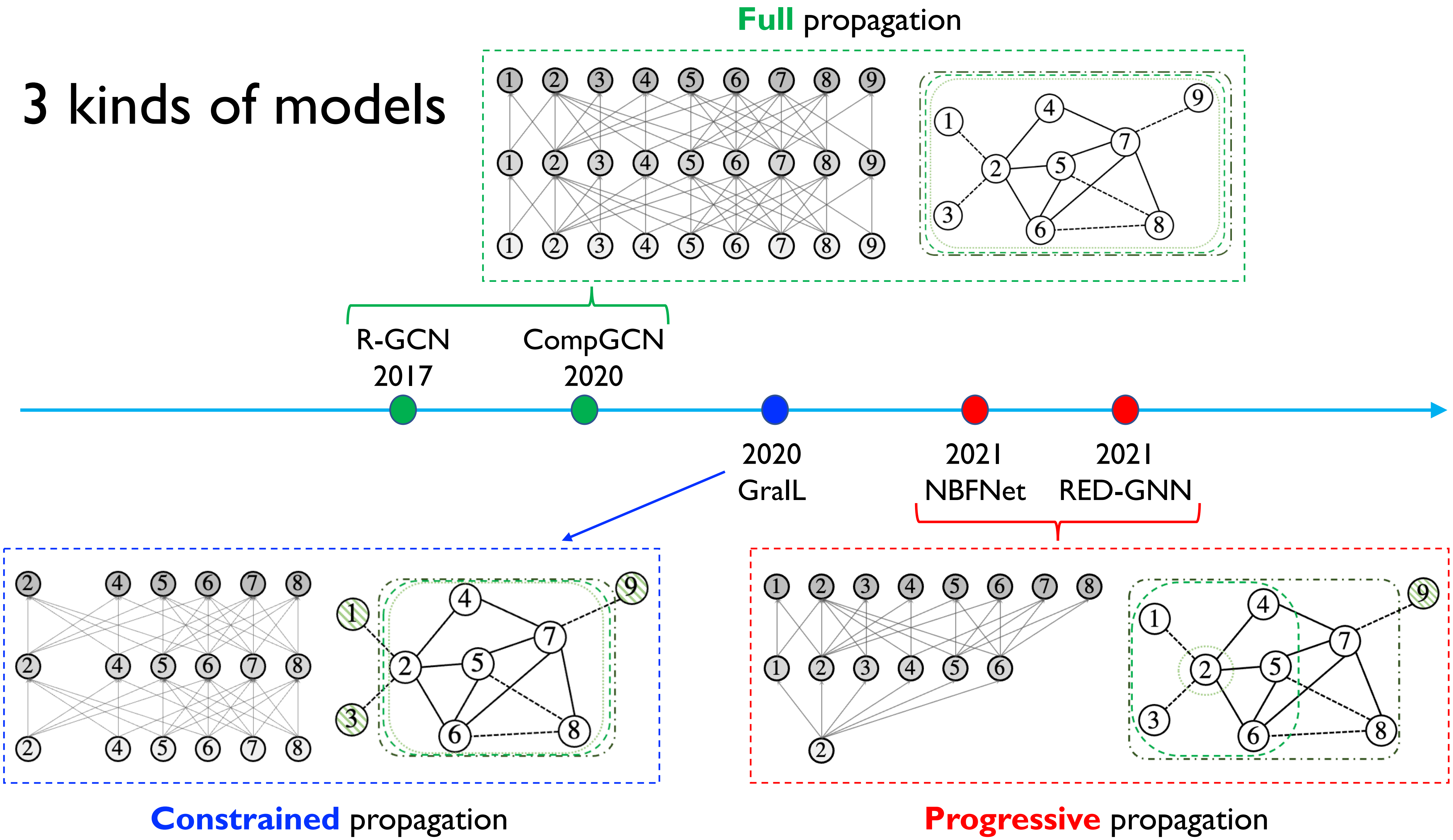
Graph neural network

$$\boldsymbol{m}^{\ell}_{(u,v)} = \mathrm{MESS}(\boldsymbol{h}^{\ell-1}_{u}, \boldsymbol{h}^{\ell-1}_{v}, \boldsymbol{e}_{uv})$$

$$\boldsymbol{h}^{\ell}_{v} = \delta\Big(\mathrm{AGG}\big(\boldsymbol{m}^{\ell}_{(u,v)}, u \in \mathcal{N}(v)\big)\Big)$$



3 kinds of models



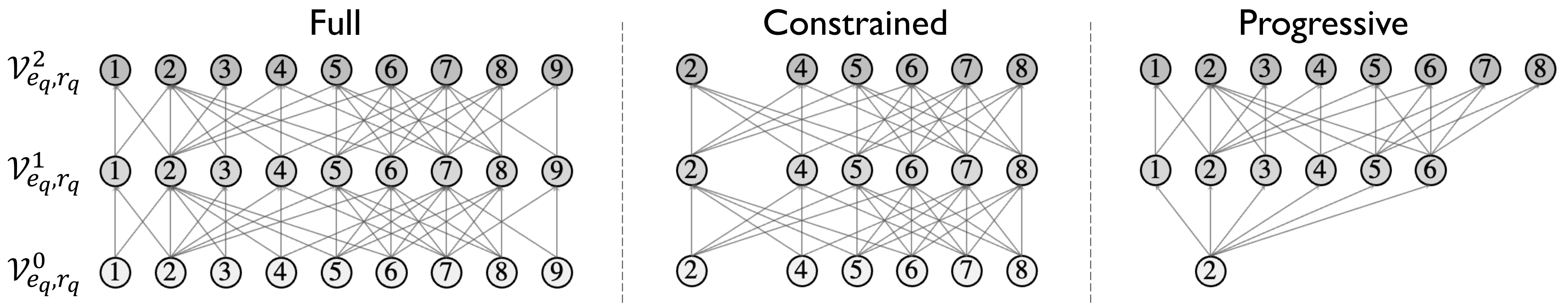
Outline

- Background
- Method
- Experiments
- Summary

Problem formulation

- Query -dependent propagation path

$\hat{\mathcal{G}}^{L}_{e_q,r_q} = \left\{\mathcal{V}^{0}_{e_q,r_q}, \mathcal{V}^{1}_{e_q,r_q}, \dots, \mathcal{V}^{L}_{e_q,r_q}\right\}$ as the sets of involved entities in each propagation step for query $(e_q, r_q, ?)$



- Problems when L is large

- large memory cost
- over-smoothing

- extremely high inference cost

- exponentially increased nodes

Challenges

- Reduce the size of propagation path through sampling

$$\widehat{\mathcal{G}}^L_{e_q,r_q} = \{\mathcal{V}^0_{e_q,r_q}, \mathcal{V}^1_{e_q,r_q}, \dots, \mathcal{V}^L_{e_q,r_q}\},$$

$$\text{s.t. } \mathcal{V}^\ell_{e_q,r_q} = \begin{cases} \{e_q\} & \ell = 0 \\ S(\mathcal{V}^{\ell-1}_{e_q,r_q}) & \ell = 1 \dots L \end{cases}.$$

- Two challenges of the sampling strategy $S(\cdot)$
  - the target answer e_a is unknown given $(e_q, r_q, ?)$ $\longrightarrow$ connection lost
  - semantic dependency is complex $\longrightarrow$ bad sampling signal

- Existing sampling approaches are not applicable

(i) no target preserving; (ii) no relation consideration; (iii) no direct supervision

The proposed method

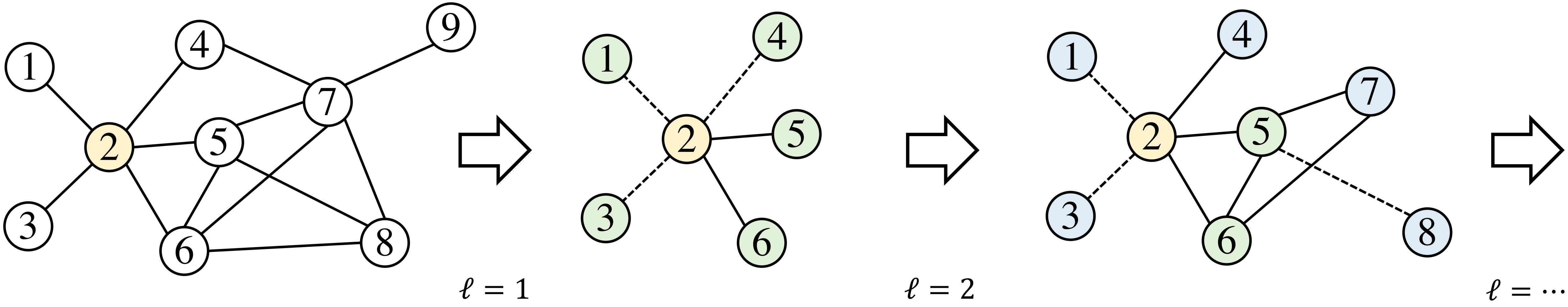
Challenges	1. connection lost	2. bad sampling signal
Proposed	connection-preserving incremental sampling	learning-based sematic aware distribution
Key idea	preserve the previous entities & sample from the newly visited ones $\mathcal{V}^0_{e_q,r_q} \subseteq \mathcal{V}^1_{e_q,r_q} \cdots \subseteq \mathcal{V}^L_{e_q,r_q}$	introduce a parameterized distribution & borrow knowledge from the GNN $\mathcal{V}^\ell_{e_q,r_q} = S(\mathcal{V}^{\ell-1}_{e_q,r_q}; \boldsymbol{\theta}^\ell)$

adaptively sample semantically relevant entities during propagation

Incremental sampling

/ / entities covered in the 0/1/2-th steps

Linear complexity!



Candidate generation:

the newly-visit neighboring entities of last step

$$\overline{\mathcal{V}}^{\ell}_{e_q,r_q} := \texttt{CAND}(\mathcal{V}^{\ell-1}_{e_q,r_q}) = \mathcal{N}(\mathcal{V}^{\ell-1}_{e_q,r_q}) \setminus \mathcal{V}^{\ell-1}_{e_q,r_q}.$$

e.g. ① ③ ④ ⑤ ⑥ when $l = 1$

① ③ ④ ⑦ ⑧ when $l = 2$

Candidate sampling:

sample K entities with replacement from candidates

$$\mathcal{V}^{\ell}_{e_q,r_q} := \mathcal{V}^{\ell-1}_{e_q,r_q} \cup \texttt{SAMP}(\overline{\mathcal{V}}^{\ell}_{e_q,r_q}).$$

e.g. ⑤ ⑥ when $l = 1$

④ ⑦ when $l = 2$

Semantic-aware distribution

- Parameterized sampling distribution:
  - Sharing the knowledge in GNN representations $\boldsymbol{h}_e^\ell$
  - Adaptive based on the learnable parameters $\boldsymbol{\theta}^\ell$

$$p^\ell(e) := \exp\left(g(\boldsymbol{h}_e^\ell; \boldsymbol{\theta}^\ell)/\tau\right) \Big/ \sum_{e' \in \overline{\mathcal{V}}_{e_q, r_q}^\ell} \exp\left(g(\boldsymbol{h}_{e'}^\ell; \boldsymbol{\theta}^\ell)/\tau\right)$$

- Learning strategy:
  - Gumbel-trick to enable backward propagation on hard samples.
    - Sampling: get top-K based on gumbel-logits

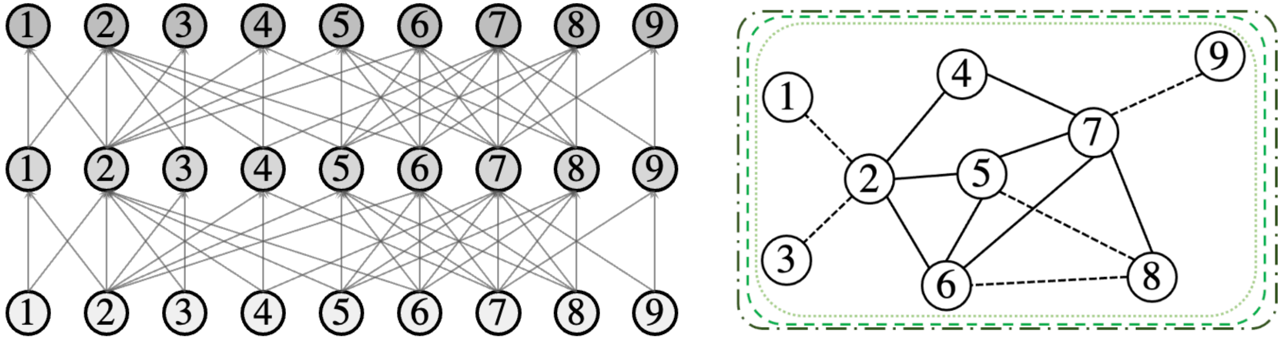
      $G_e := g\left(\boldsymbol{h}_e^\ell; \boldsymbol{\theta}^\ell\right) - \log(-\log U_e)$ with $U_e \sim \mathrm{Uniform}(0,1)$ for the candidate entities
    - Enable backpropagation: straight-through estimation

      $\boldsymbol{h}_e^\ell = \left(\boxed{1 - \text{no\_grad}\left(p^\ell(e)\right) + p^\ell(e)}\right) \cdot \boldsymbol{h}_e^\ell$ for the selected entities

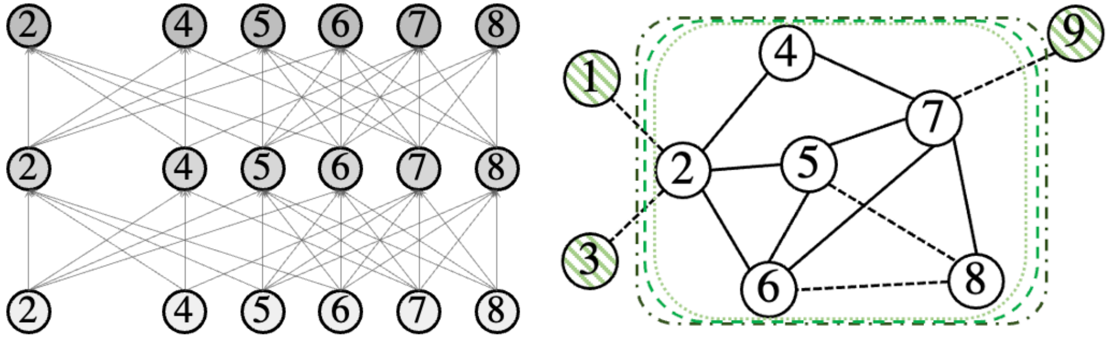
Has no influence during forward propagation,
but provides gradient for θ^ℓ when backward.

Overall comparison

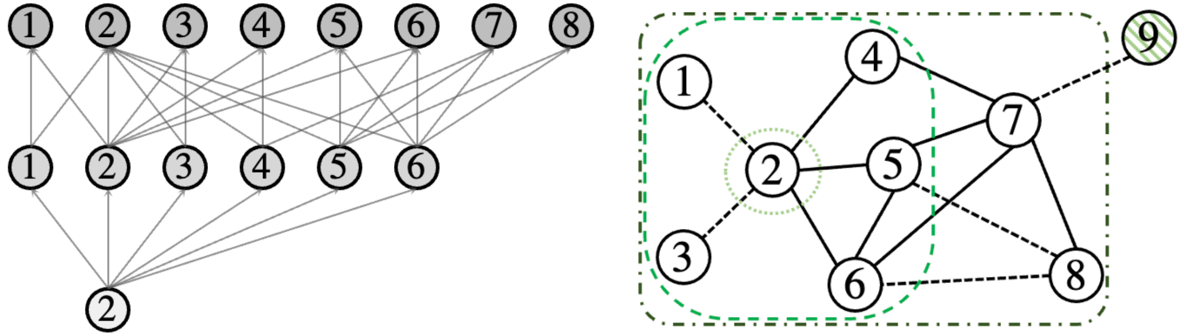
Full propagation



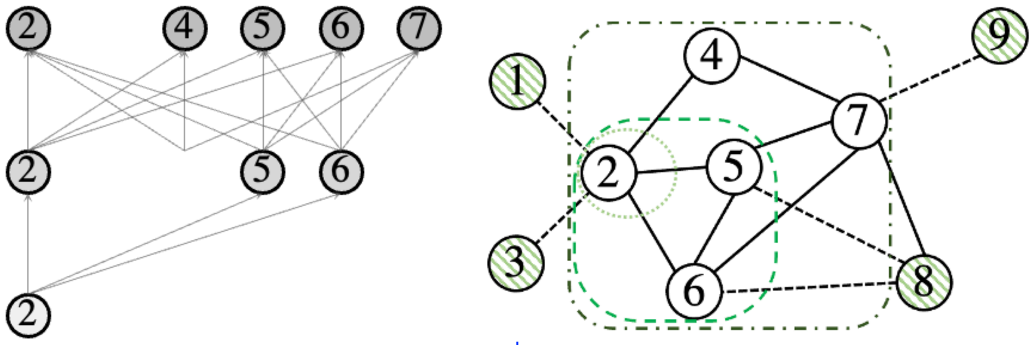
Constrained propagation



Progressive propagation



AdaProp



Achieves smaller complexity, deeper steps and query-dependent.

Outline

- Background
- Method
- Experiments
- Summary

Experiments | Quantitative Results

type	models	Family			UMLS			WN18RR			FB15k237			NELL-995			YAGO3-10		
		MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
non-GNN	ConvE	0.912	83.7	98.2	0.937	92.2	96.7	0.427	39.2	49.8	0.325	23.7	50.1	0.511	44.6	61.9	0.520	45.0	66.0
	QuatE	0.941	89.6	99.1	0.944	90.5	99.3	0.480	44.0	55.1	0.350	25.6	53.8	0.533	46.6	64.3	0.379	30.1	53.4
	RotatE	0.921	86.6	98.8	0.925	86.3	99.3	0.477	42.8	57.1	0.337	24.1	53.3	0.508	44.8	60.8	0.495	40.2	67.0
	MINERVA	0.885	82.5	96.1	0.825	72.8	96.8	0.448	41.3	51.3	0.293	21.7	45.6	0.513	41.3	63.7	–	–	–
	DRUM	0.934	88.1	99.6	0.813	67.4	97.6	0.486	42.5	58.6	0.343	25.5	51.6	0.532	46.0	66.2	0.531	45.3	67.6
	RNNLogic	0.881	85.7	90.7	0.842	77.2	96.5	0.483	44.6	55.8	0.344	25.2	53.0	0.416	36.3	47.8	0.554	50.9	62.2
	RLogic	–	–	–	–	–	–	0.47	44.3	53.7	0.31	20.3	50.1	–	–	–	0.36	25.2	50.4
GNNs	CompGCN	0.933	88.3	99.1	0.927	86.7	99.4	0.479	44.3	54.6	0.355	26.4	53.5	0.463	38.3	59.6	0.421	39.2	57.7
	NBFNet	0.989	**98.8**	98.9	0.948	92.0	**99.5**	0.551	49.7	66.6	0.415	32.1	**59.9**	0.525	45.1	63.9	0.550	47.9	68.6
	RED-GNN	**0.992**	**98.8**	**99.7**	0.964	94.6	99.0	0.533	48.5	62.4	0.374	28.3	55.8	0.543	47.6	65.1	0.559	48.3	68.9
	AdaProp	0.988	98.6	99.0	**0.969**	**95.6**	**99.5**	**0.562**	**49.9**	**67.1**	**0.417**	**33.1**	58.5	**0.554**	**49.3**	**65.5**	**0.573**	**51.0**	**68.5**

Evaluation with transductive settings

metric	methods	WN18RR				FB15k237				NELL-995			
		V1	V2	V3	V4	V1	V2	V3	V4	V1	V2	V3	V4
Hit@10 (%)	RuleN	73.0	69.4	40.7	68.1	44.6	59.9	60.0	60.5	76.0	51.4	53.1	48.4
	Neural LP	77.2	74.9	47.6	70.6	46.8	58.6	57.1	59.3	87.1	56.4	57.6	53.9
	DRUM	77.7	74.7	47.7	70.2	47.4	59.5	57.1	59.3	87.3	54.0	57.7	53.1
	GraIL	76.0	77.6	40.9	68.7	42.9	42.4	42.4	38.9	56.5	49.6	51.8	50.6
	CoMPILE	74.7	74.3	40.6	67.0	43.9	45.7	44.9	35.8	57.5	44.6	51.5	42.1
	NBFNet	82.7	79.9	56.3	70.2	51.7	63.9	58.8	55.9	79.5	63.5	60.6	59.1
	RED-GNN	79.9	78.0	52.4	72.1	48.3	62.9	60.3	62.1	86.6	60.1	59.4	55.6
	AdaProp	**86.6**	**83.6**	**62.6**	**75.5**	**55.1**	**65.9**	**63.7**	**63.8**	**88.6**	**65.2**	**61.8**	**60.7**

Evaluation with inductive settings

AdaProp achieves the **state-of-the-art performance** in both transductive and inductive KG reasoning settings.

Experiments | Quantitative Results

Table 4: Comparison of different sampling methods.

learn	methods	WN18RR EI(L)	WN18RR ToE(L)	WN18RR MRR	FB15k237-v1 EI(L)	FB15k237-v1 ToE(L)	FB15k237-v1 Hit@10
not learned	Node-wise	4831	1.38E−4	.416	585	1.35E−3	38.9
	Layer-wise	5035	1.46E−4	.428	554	1.45E−3	37.2
	Subgraph	5098	1.57E−4	.461	578	1.50E−3	40.5
	Incremental	4954	1.61E−4	.472	559	1.52E−3	40.1
learned	Node-wise	4913	1.52E−4	.529	561	1.47E−3	50.4
	Layer-wise	4871	1.64E−4	.533	556	1.55E−3	52.4
	Incremental	4749	1.78E−4	.562	564	1.57E−3	55.1

The **incremental sampling** is better than the other sampling strategies.

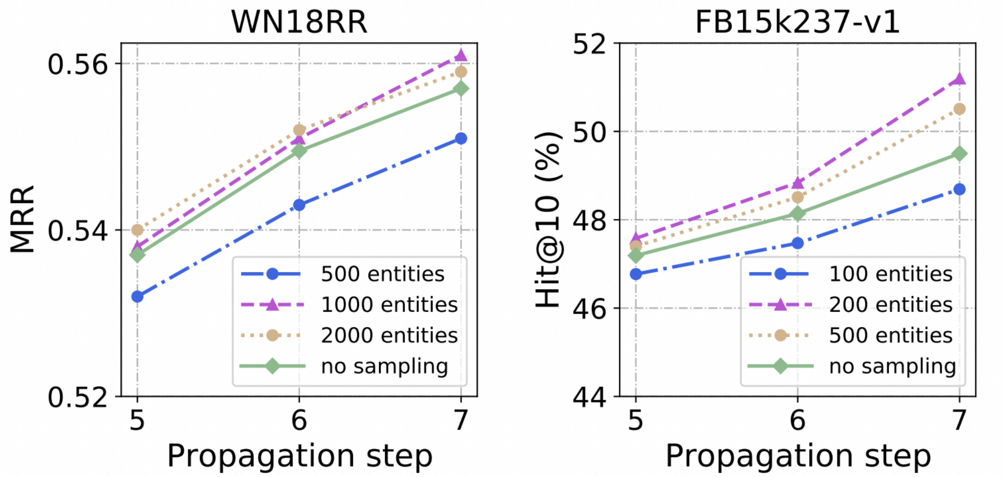
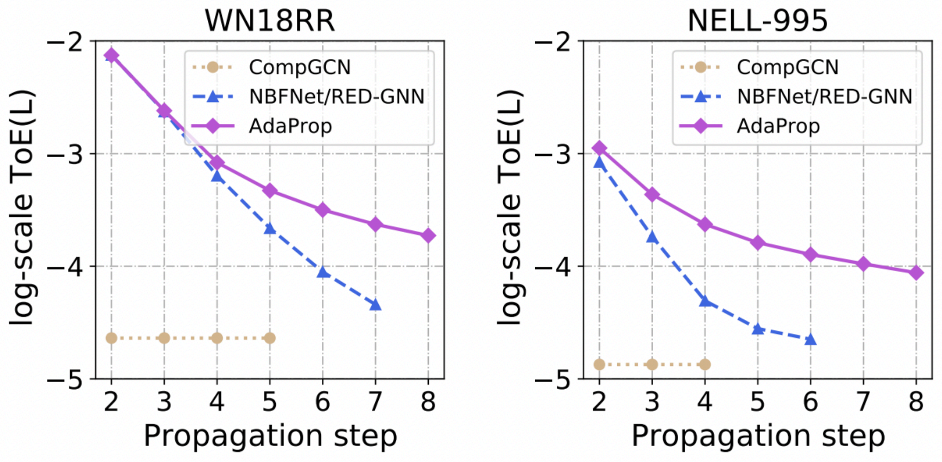


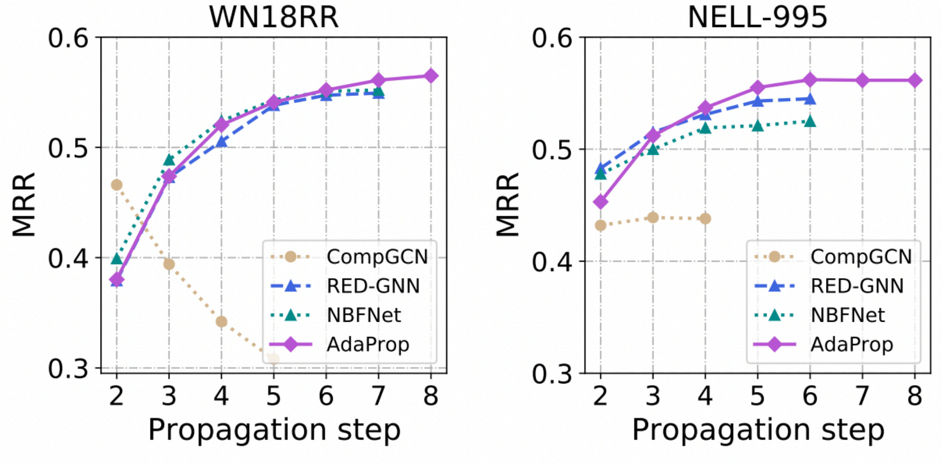
Figure 6: Ablation study with K on different L. Each line represents the performance of a given K with different L.

The performance gains by **sampling more entities** would gradually become marginal or even worse.

Experiments | Qualitative Results



(a) ToE(L)



(b) Testing performance

Figure 3: Comparison of GNN-based methods w.r.t. L.

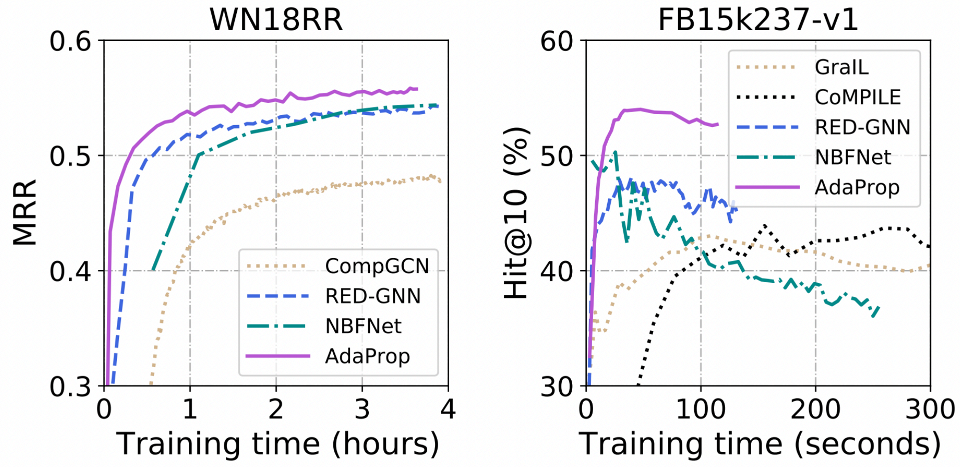


Figure 4: Learning curves of different GNN-based methods.

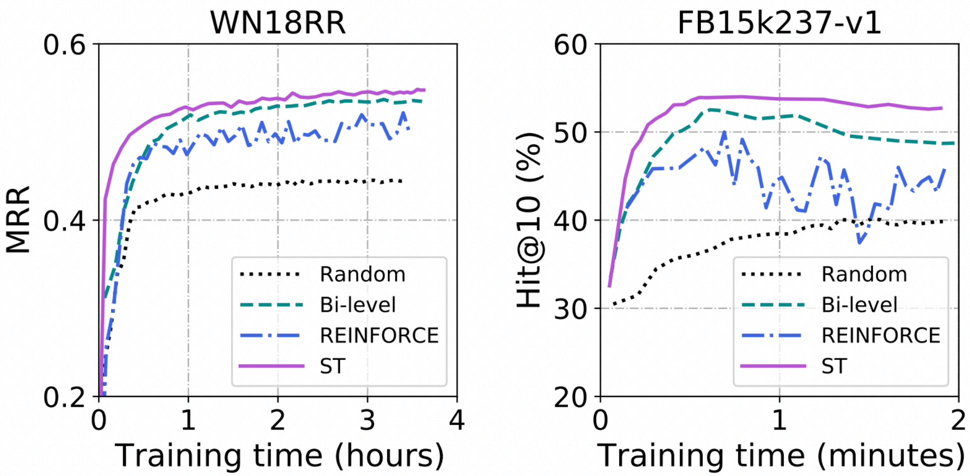
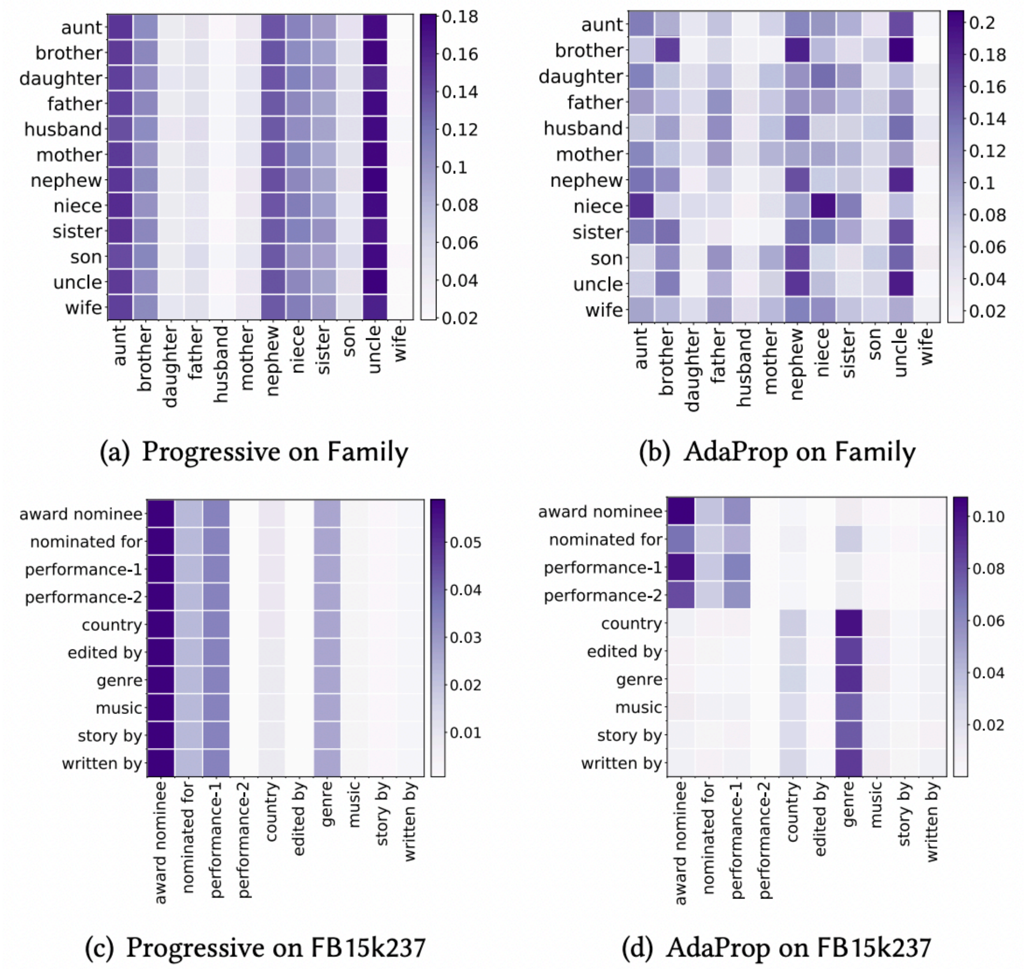


Figure 5: Comparison of learning strategies for the sampler.

Experiments | Qualitative Results

semantic-aware

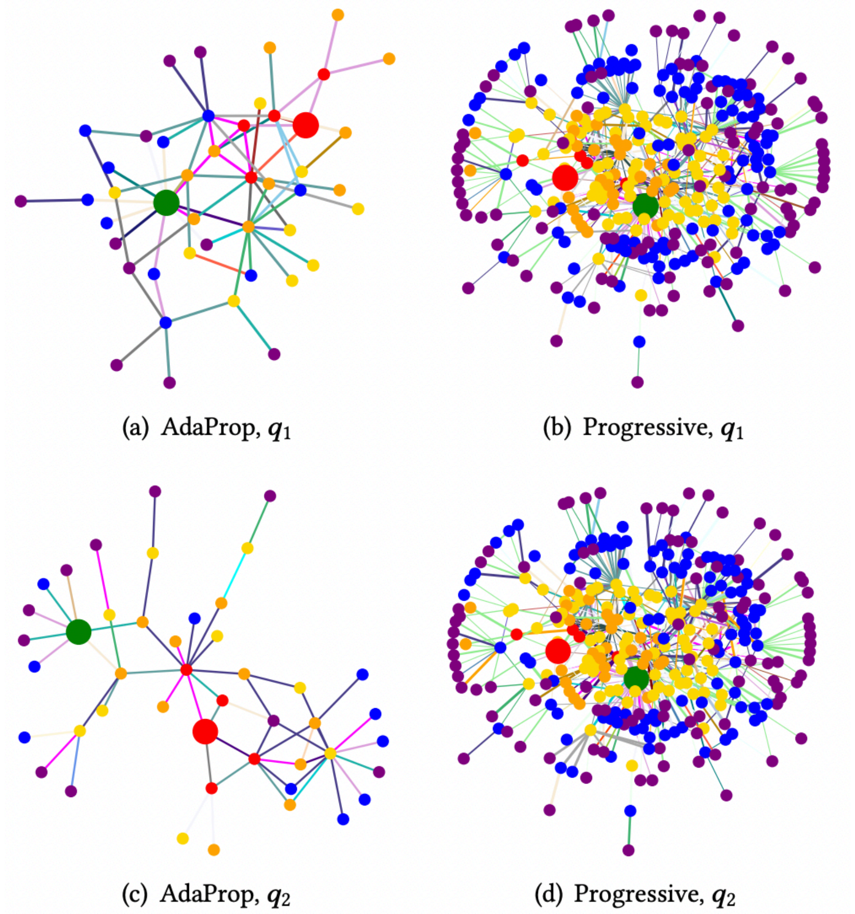


(a) Progressive on Family (b) AdaProp on Family

(c) Progressive on FB15k237 (d) AdaProp on FB15k237

Heatmaps of relation type ratios in the propagation path

connection-preserving



(a) AdaProp, q_1 (b) Progressive, q_1

(c) AdaProp, q_2 (d) Progressive, q_2

Exemplar propagation paths on FB15k237-v1 dataset

Outline

- Background
- Method
- Experiments
- Summary

Summary

Three major contributions:

- We propose an **incremental sampling scheme**
  - only has **linear complexity** with regard to the propagation steps
  - can **preserve the layer-wise connections** between sampled entities
- We design a semantic-aware **Gumbel top-k distribution**
  - can **adaptively select local neighborhoods** relevant to the query relation
  - **learned** by a straight through estimator
- We achieve the **state-of-the-art** performance
  - in both **transductive and inductive** KG reasoning settings
  - case study shows that the learned sampler is **query-dependent** and **semantic-aware**

Thanks for your listening

Discussion: yzhangee@connect.ust.hk
cszkzhou@comp.hkbu.edu.hk