Neural-Symbolic Reasoning Over Knowledge Graph

Kewei Cheng
Advisor: Prof. Yizhou Sun
Knowledge Graphs

- A knowledge graph is a collection of real-world facts.
- They enable many downstream applications (NLP tasks, QA systems, etc)
Knowledge Graph Reasoning

- Knowledge graphs are usually incomplete
- A fundamental task: predicting missing links (or facts) by reasoning on existing facts
Symbolic Methods & Neural Methods

Traditional Symbolic Reasoning

liveIn (Mina Miller, USA)
officialLanguage (USA, English)

Country = USA
Language = English

speakLanguage(Person, Language) ← liveIn(Person, Country) ∧ officialLanguage (Country, Language)

Logical Inference

speakLanguage (Mina Miller, English)
New fact

Observed Facts

Definite Horn rule

Modern Representation Learning

Thomas Alva Edison

isMarriedTo

Mary Stilwell

isMarriedTo

Mina Miller

liveIn

USA

IsMarriedTo

liveIn

English

officialLanguage

Embedding Learning

Predict
Neural-Symbolic Reasoning

Leverage the advantages of both neural and symbolic reasoning for knowledge graph reasoning

- **Symbolic**: ability of using domain knowledge, interpretability
- **Neural**: efficiency, capacity

**One of the Challenges**: Domain knowledge is encoded as logical rules, logical rules are usually need to be specified by hand
Logical Rule Induction/Learning

- Given: a background KG $g$
- Goal: learn weighted **chain-like Horn rule** of the following form

$$\alpha : r_h(x, y) \leftarrow r_{b_1}(x, z_1) \land \cdots \land r_{b_n}(z_{n-1}, y)$$

where $\alpha \in [0,1]$ is the confidence score associated with this rule, indicating how likely the rule holds true.
GAP Between Instance Rules and Template Rules

Instance Rules
\[
\begin{align*}
\text{hasGrandMother}(\text{Amy}, \text{Cara}) & \equiv \text{hasMother}(\text{Amy}, \text{Bess}) \land \text{hasMother}(\text{Bess}, \text{Cara}) \\
\text{hasGrandMother}(\text{Bess}, \text{Dana}) & \equiv \text{hasMother}(\text{Bess}, \text{Cara}) \land \text{hasMother}(\text{Cara}, \text{Dana}) \\
\text{hasGrandMother}(\text{Cara}, \text{Eva}) & \equiv \text{hasMother}(\text{Cara}, \text{Dana}) \land \text{hasMother}(\text{Dana}, \text{Eva})
\end{align*}
\]

Template Rule
\[
\text{hasGrandMother}(x, y) \equiv \text{hasMother}(x, z) \land \text{hasMother}(z, y)
\]

How to bridge the gap between instance level observation and schema level abstraction?
Previous Works

- **Rely on observed rule instances to evaluate the plausibility of logical rules.**
  - Limited scalability to KG size
  - May not be reliable due to the widely existing missing facts in KGs

- **Ignore deductive nature of logical rules**
  - i.e., the ability to recombine known parts and rules to form new sequences while reasoning over relational data
  - e.g.,

```
Question  ?← hasMother(x, z₁) ∧ hasMother(z₁, z₂) ∧ hasSon(z₂, y)
```

Diagram:

```
hasGrandMother(x, z₁)
```

```
hasUncle(x, z₁)
```

```
hasMother(x, z₁) ∧ hasMother(z₁, z₂) ∧ hasSon(z₂, y)
```
Our Proposed Method: RLogic

- Propose a new measure for rule evaluation based on the probability that the rule body can be replaced by the rule head

\[ q(r_h = r_i | r_b) \]

approximate

Confidence: 3/4

\[ \text{hasGrandMother}(x, y) \iff \text{hasMother}(x, z) \land \text{hasMother}(z, y) \]
How to Incorporate Inductive Nature

• A representation-learning based model can be used to learn \( q(r_h = r_i | r_b) \)
  
  E.g, a sequential model, such as RNN

  However, RNN directly models the entire sequence length without explicitly capturing the deductive nature of logical rule

How to incorporate inductive nature to break the learning into recursive process?
Relation Path Encoder

- Basic idea: push deductive reasoning into rule learning
  - reduce a long relation path \([r_{b_1}, r_{b_2}, \ldots, r_{b_n}]\) by replacing the relation pair \([r_{b_i}, r_{b_{i+1}}]\) in relation path with their head \(r_h\) recursively until the relation path being transformed into a single head

- E.g., given a relation path \([r_{b_1}, r_{b_2}, r_{b_3}]\)
  
  \[
  q(r_h|r_{b_1}, r_{b_2}, r_{b_3}) = \sum_k q(r_h|r_k, r_{b_3})q(r_k|r_{b_1}, r_{b_2})
  \]
Our Proposed Method: RLogic

• Two components
  
  1. **Relation path encoder**: reduces a relation path $r_b$ into a single head $r_h$ by recursively merge relation pairs in $r_b$

  2. **Close ratio predictor**: bridges the gap between “ideal prediction” following logical rules and “real observation” by predicting the ratio that the relation path $r_b$ will close.

• Training process
  
  Sample closed path instances and maximize their likelihood
Future Works

• Extend the RLogic to learn rules directly from unstructured natural language

• **Challenges:** relations are not directly provided as a graph

• **Possible solution:** having an end-to-end differentiable encoder for producing the fact embeddings conditioned on the text
Thank you!

Q & A