UCLA Samueli School of Engineering

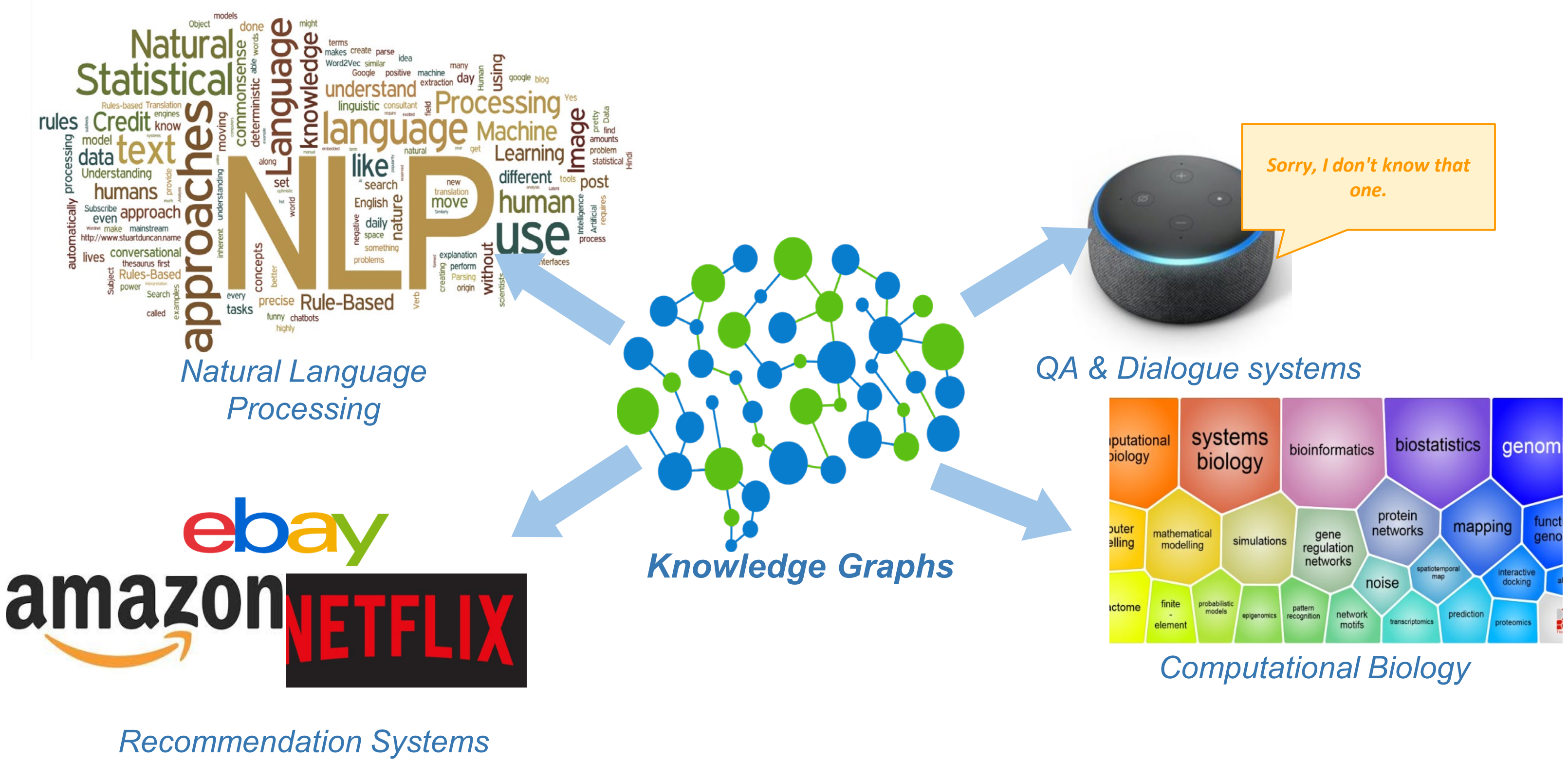
Neural-Symbolic Reasoning Over Knowledge Graph

Kewei Cheng

Advisor: Prof. Yizhou Sun

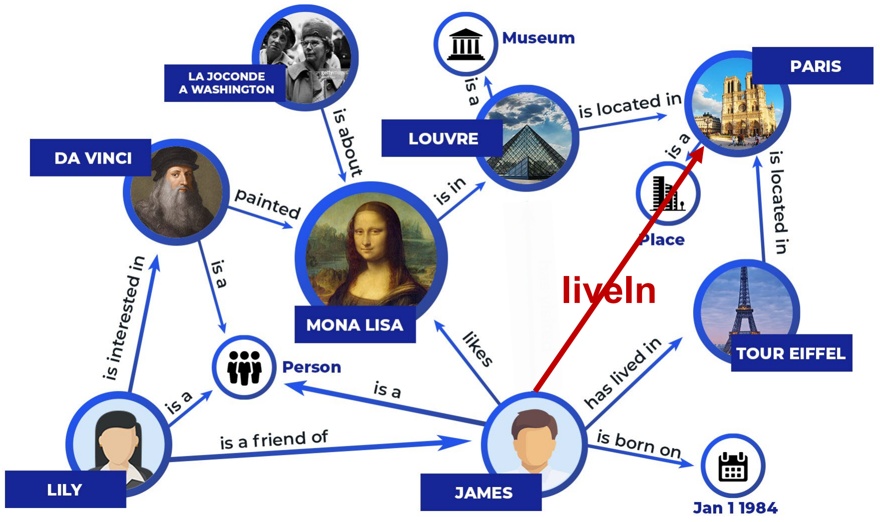
Knowledge Graphs

- 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



Knowledge Graph

Symbolic Methods & Neural Methods

Traditional Symbolic Reasoning

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

Observed Facts

Country = USA
Language = English

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

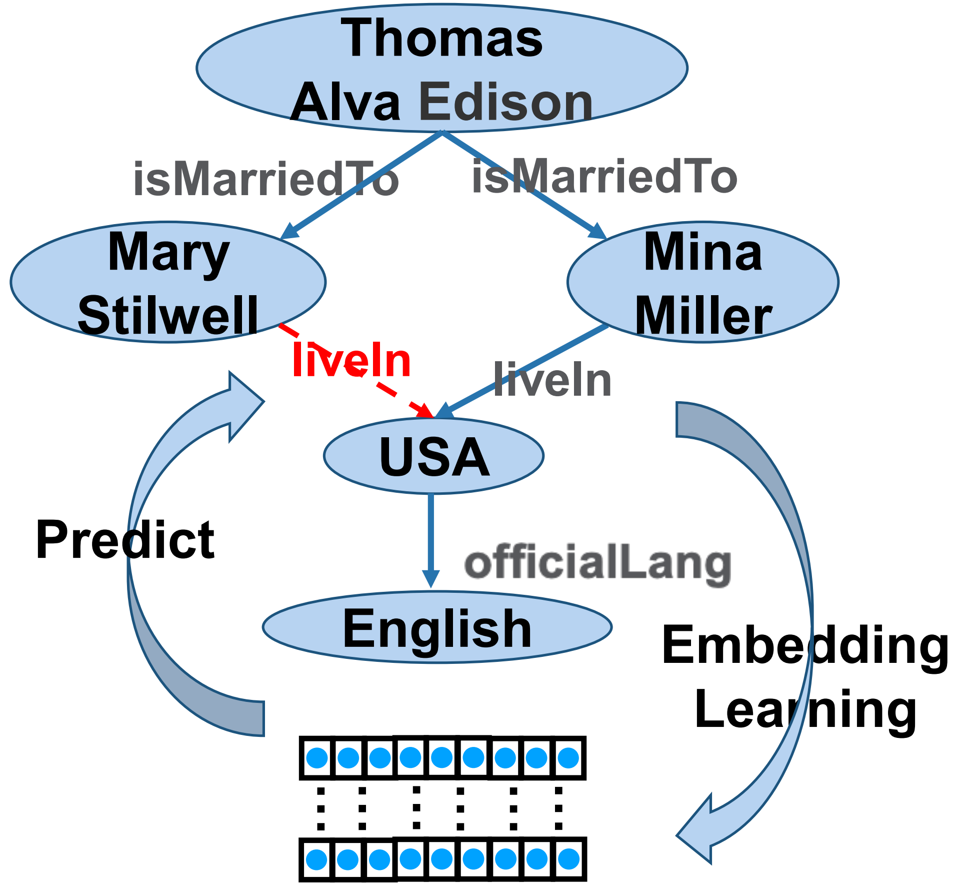
Definite Horn rule

Logical Inference

speakLanguage (Mina Miller, English)

New fact

Modern Representation Learning



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

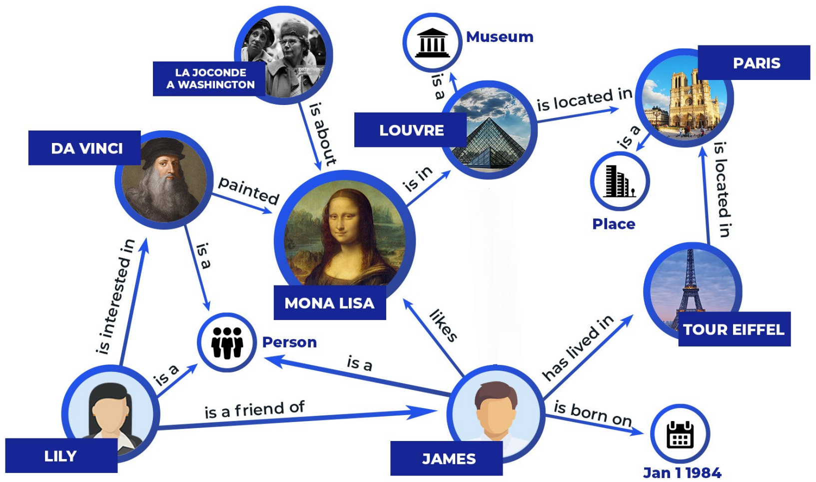
One of the Challenges: Domain knowledge is encoded as logical rules, logical rules are usually need to be specified by hand

Logical Rule Induction/Learning

- Given: a background KG $\mathcal{G}$
- Goal: learn weighted **chain-like Horn rule** of the following form

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

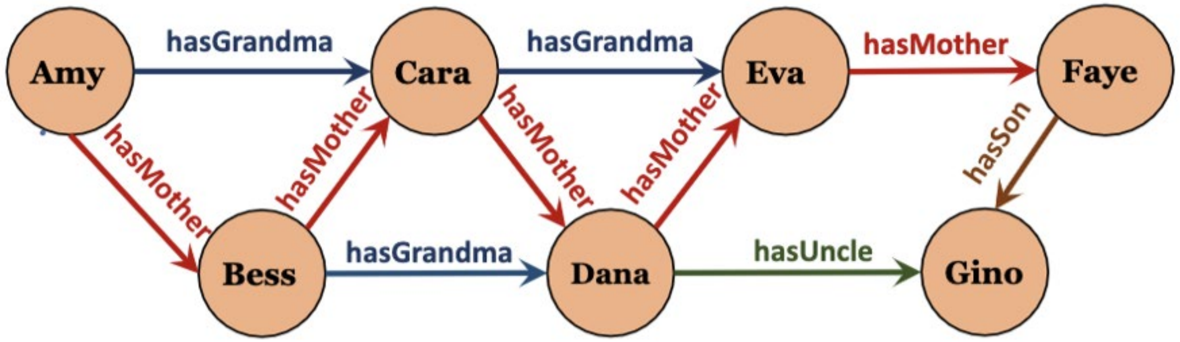
where α∈[0,1] is the confidence score associated with this rule, indicating how likely the rule holds true.



liveIn(a,c) ⇐ liveIn(a,b) ∧ isLocatedIn(b,c)

Logical Rules

GAP Between Instance Rules and Template Rules



Instance Rules

hasGrandMother(Amy, Cara) ⇐ hasMother(Amy, Bess) ∧ hasMother(Bess, Cara)
hasGrandMother(Bess, Dana) ⇐ hasMother(Bess, Cara) ∧ hasMother(Cara, Dana)
hasGrandMother(Cara, Eva) ⇐ hasMother(Cara, Dana) ∧ hasMother(Dana, Eva)

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

Template Rule

hasGrandMother(x, y) ⇐ hasMother(x, z) ∧ hasMother(z, y)

UCLA Samueli School of Engineering

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** $? \Leftarrow \text{hasMother}(x, z_1) \wedge \text{hasMother}(z_1, z_2) \wedge \text{hasSon}(z_2, y)$

$$\text{hasMother}(x, z_1) \wedge \text{hasMother}(z_1, z_2) \wedge \text{hasSon}(z_2, y)$$

$$\text{hasGrandMother}(x, z_1)$$

$$\text{hasUncle}(x, z_1)$$

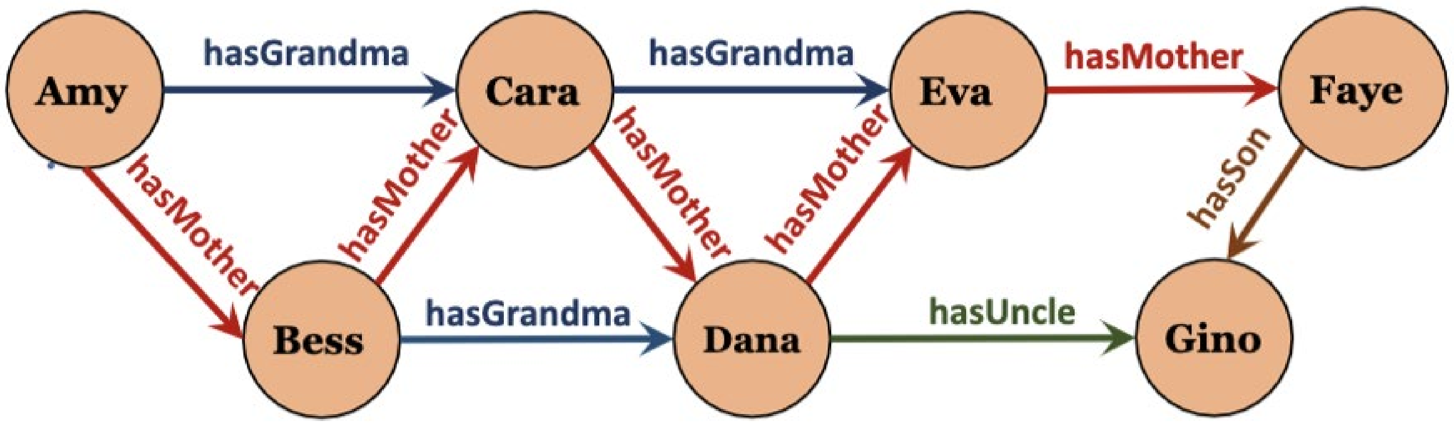
Our Proposed Method: RLogic

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

$$q(r_h = r_i | \mathbf{r_b})$$

approximate

Confidence: 3/4



hasGrandMother(x, y) ⇐ hasMother(x, z) ∧ hasMother(z, y)

How to Incorporate Inductive Nature

- A representation-learning based model can be used to learn $\mathrm{q}(r_h = r_{\mathrm{i}} | \boldsymbol{r}_{\mathbf{b}})$
  - 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

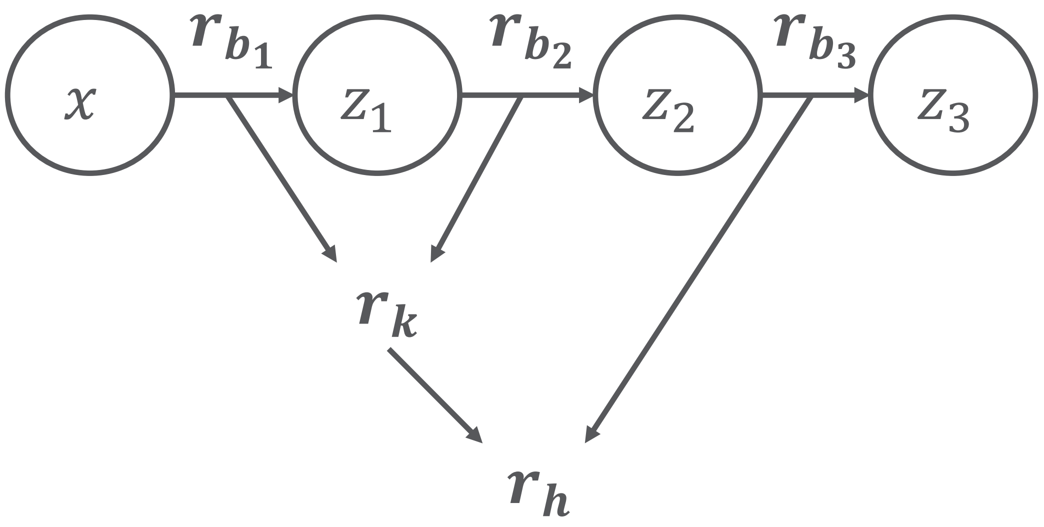
How to incorporate inductive nature to
break the learning into recursive process?

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

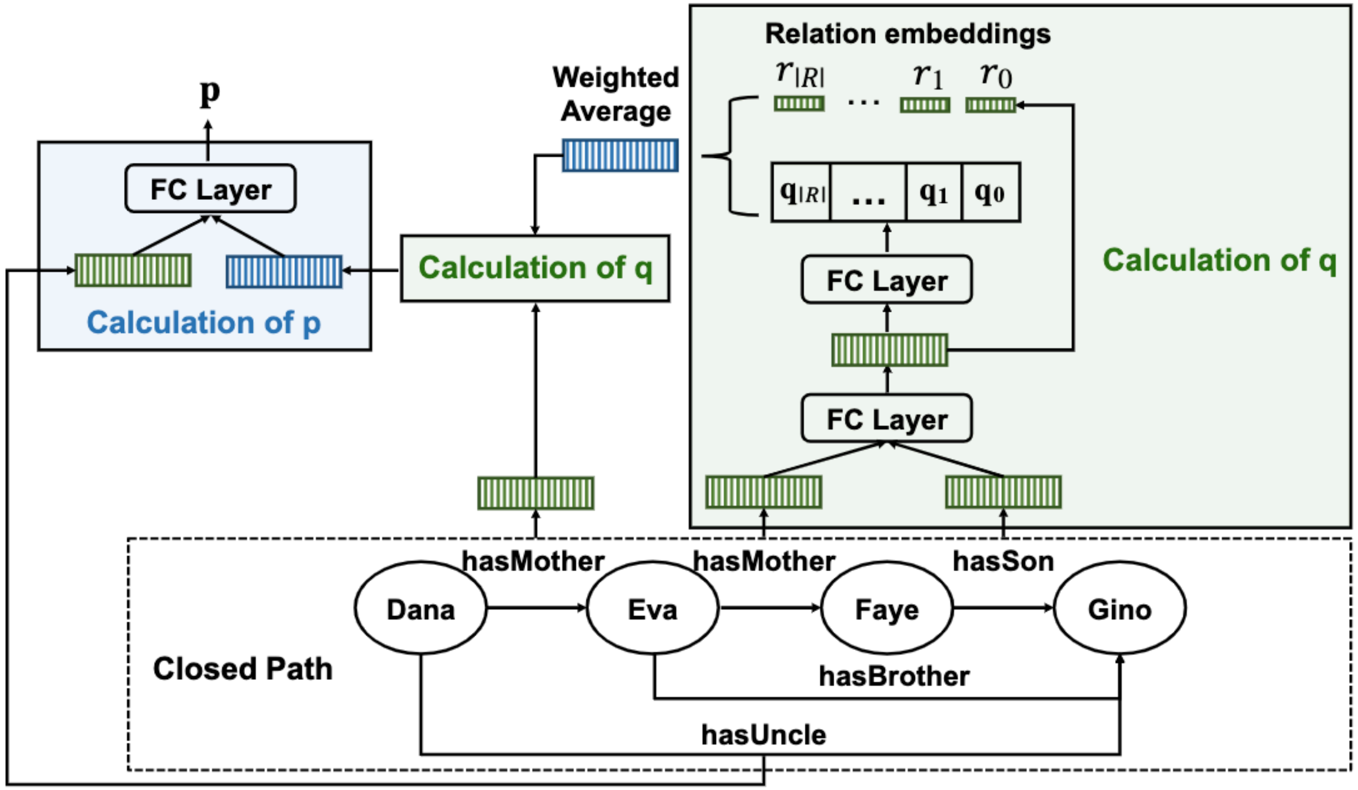
- E.g., given a relation path $[r_{b_1}, r_{b_2}, r_{b_3}]$

$$q(r_h | r_{b_1}, r_{b_2}, r_{b_3}) = \sum_k q(r_h | r_k, r_{b_3}) q(r_k | r_{b_1}, r_{b_2})$$



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 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