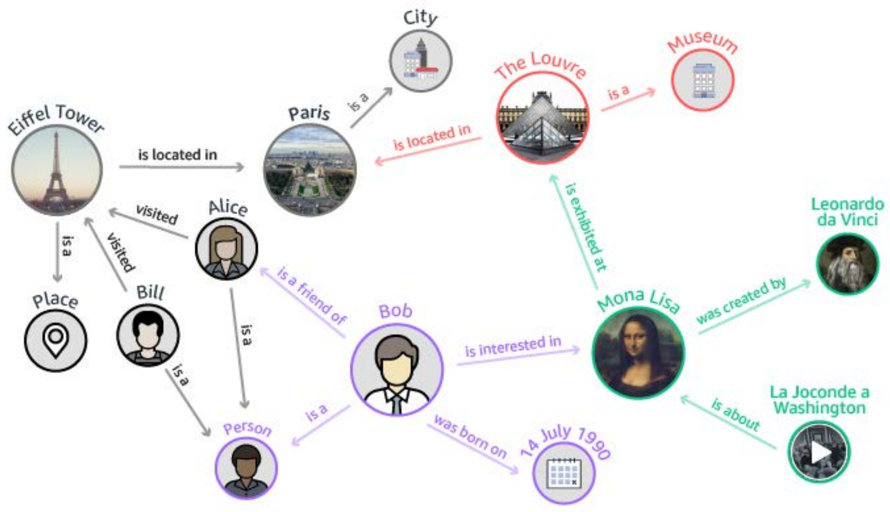
UCLA Samueli Computer Science

Fuzzy Logic based Logical Query Answering on Knowledge Graphs

Xuelu Chen, Ziniu Hu, Yizhou Sun

What are knowledge graphs?

- Multi-relational graph data
- Provide structured representation for semantic relationships between real-world entities
- Serve as important knowledge sources for many real-world applications
    - Question answering systems
    - Dialogue agents
    - Recommender systems
    - …



A triple $p(s, o)$ represents a fact, e.g.,

$locatedIn(\textbf{Eiffel Tower}, \textbf{Paris})$

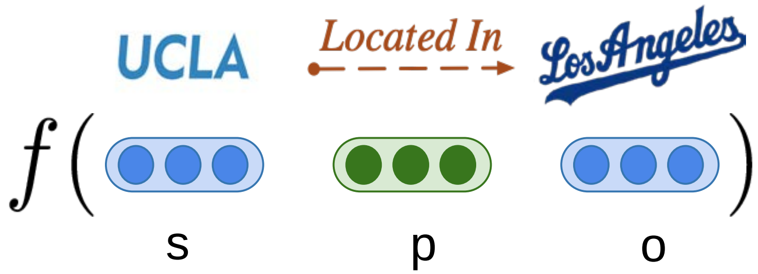
predicate subject object

Knowledge graph queries

- **Atomic Query (Knowledge Graph Completion)**
  - Given an incomplete triple, infer the missing entity
    - E.g., `LocatedIn(UCLA, ?)`

Triple

Embedding based Scoring Function



  - Related work:
    - TransE [Bordes et al. 2013], DistMult [Yang et al. 2015]; ComplEx [Trouillon et al. 2016]; RotatE [Sun et al. 2019], ...

- **More Complex Logical Query**
  - Given a more complicated query, infer the entity
    - E.g., *"Return singers who have sung songs that were written by John Lennon or Paul McCartney but never won Grammy Award"*

Knowledge graph queries

"Return singers who have sung songs that were written by Lennon or McCartney but never won Grammy Award"

Logical Query

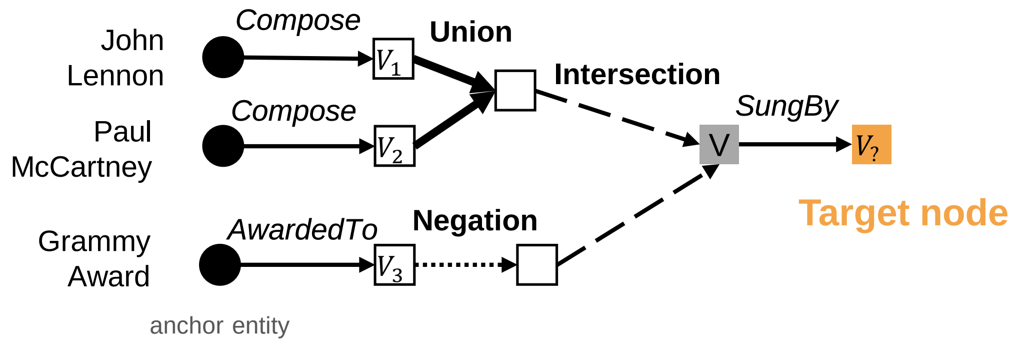
Logical Operators

$$q = V_? : \exists V \; \big(Compose(\text{John Lennon}, V) \vee Compose(\text{Paul McCartney}, V) \big)$$
$$\wedge \neg AwardedTo(\text{Grammy Award}, V)$$
$$\wedge SungBy(V, V_?)$$

Target Variable (singers)

Intermediate Variable (songs)

Computation Graph
(A Directed Acyclic Graph)



Challenging questions

- **Combining Representation Learning with Logical Reasoning**
    - How to represent an entity?
    - How to represent a set from a subquery?
    - How to define an embedding-based function denoting an entity belonging to a set?
    - How to recursively define embedding for each logical expression?
    - How to define an embedding-based function for each logical operator (and, or, negation)?
    - How to preserve logical laws (additional constraints) that logical operators should preserve?
        - Commutative, associative, etc.
    - How to train the model when additional Query-Answer pairs are not available?

Our Approach (FuzzQE)

- Bridging set and logical expressions
- Representing set, element, and membership
- Defining set operations that preserve logical laws
- Self-supervised training

Bridging set and logical expressions

- **A FOL query corresponds to an answer set (a set of entities)**

"Return singers that have sung songs that were written by Lennon or McCartney but never won Grammy Award"

Logical Query $q = V_? : \exists V \; (\mathit{Compose}(\text{John Lennon}, V) \vee \mathit{Compose}(\text{Paul McCartney}, V))$
$\wedge \neg \mathit{AwardedTo}(\text{Grammy Award}, V)$
$\wedge \mathit{SungBy}(V, V_?)$

Target Variable
(singers)

Entity Set (Answer Set) $s(x) := \exists y \; (\mathit{Compose}(\text{John Lennon}, y) \vee \mathit{Compose}(\text{Paul McCartney}, y))$
$\wedge \neg \mathit{AwardedTo}(\text{Grammy Award}, y) \wedge \mathit{SungBy}(y, x)$

$$\boxed{q := \{x \mid s(x) \; is \; true\}}$$

Query2Box[Ren, Hu, and Leskovec 2020]

Bridging set and logical expressions

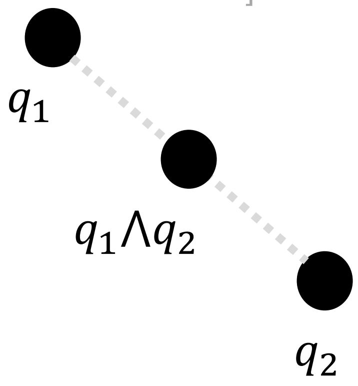
- Query Conjunction – Set Intersection
- Query Disjunction – Set Union
- Query Negation – Set Complement

Representing Set, Element, and Membership

- **Existing approaches**
  - $q, e, p(q|e)$?

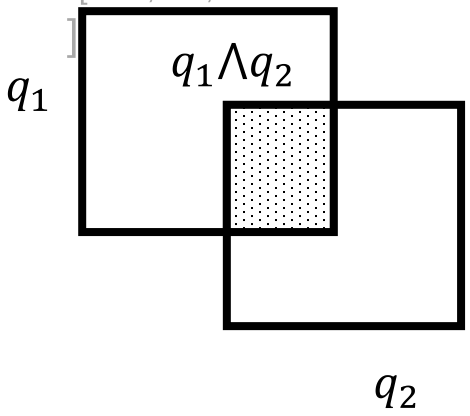
GQE

[Hamilton et al. 2018]



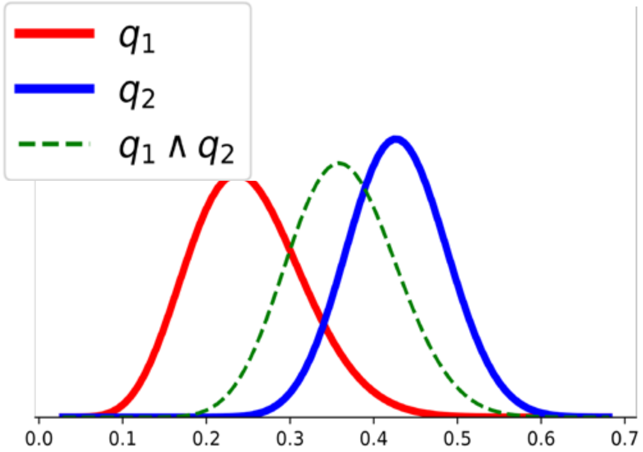
Query2Box

[Ren, Hu, and Leskovec 2020]



BetaE

[Ren and Leskovec 2020]

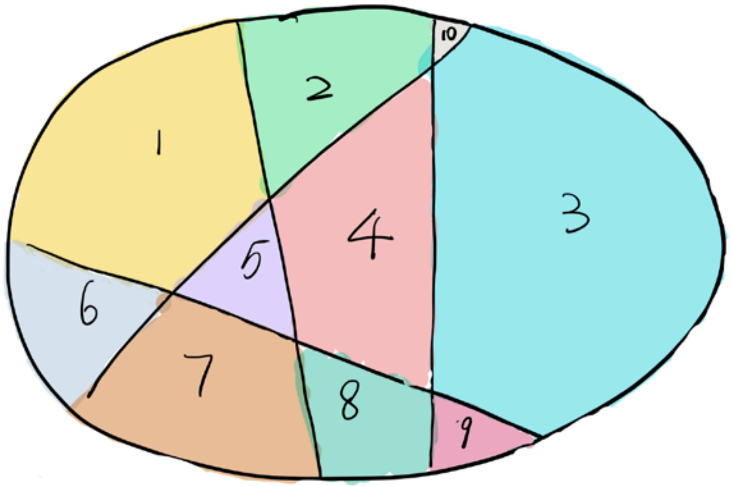


Representing Set, Element, and Membership

- **Query**
  - Query as a fuzzy set, which is represented by
    - $\boldsymbol{S}_q \in [0,1]^d$ (fuzzy vector)
    - Properties
      - $(1, \ldots, 1): \Omega$
      - $(0, \ldots, 0): \emptyset$
      - Subset, negation
- **Entity**
  - Entity as a stochastic vector
    - $\boldsymbol{P}_e \in [0,1]^d, and\ \sum_i \boldsymbol{P}_e(i) = 1$
- **Membership**
  - Embedding-based membership function

$$\phi(q, e) = \mathbb{E}_{e \sim \mathbf{p}_e}[e \in S_q] = \sum_{i=1}^{d} \Pr(e \in U_i) \Pr(U_i \subseteq S_q) = \mathbf{S}_q{}^{\intercal}\mathbf{p}_e$$

Universe Set Ω



d=10

Defining set operations that preserve logical laws

- Representing relation projection

  e.g., $Compose(\text{John Lennon}, V)$

$$\boxed{\mathbf{S}_q = \mathcal{P}_r(\mathbf{p}_e) = \mathbf{g}(\mathrm{LN}(\mathbf{W}_r\mathbf{p}_e + \mathbf{b}_r))}$$

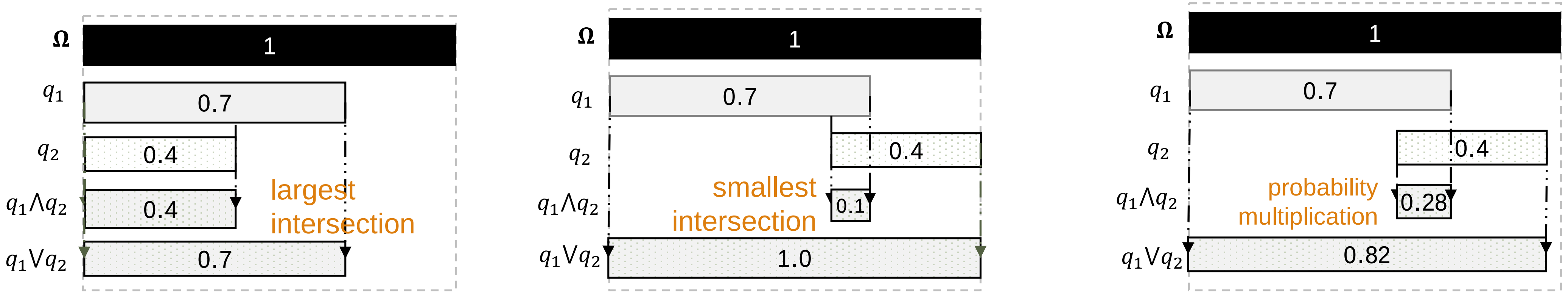
$$\mathbf{W}_r = \sum_{j=1}^{K} \alpha_{rj}\mathbf{M}_j; \quad \mathbf{b}_r = \sum_{j=1}^{K} \alpha_{rj}\mathbf{v}_j$$

RGCN [Schlichtkrull et al. 2018]

- Given the representations of two subqueries, define logical operators (set operations) with reference to product logic

$$\begin{aligned} q_1 \wedge q_2 : \quad & \mathcal{C}(\mathbf{S}_{q1}, \mathbf{S}_{q2}) = \mathbf{S}_{q1} \circ \mathbf{S}_{q2} \\ q_1 \vee q_2 : \quad & \mathcal{D}(\mathbf{S}_{q1}, \mathbf{S}_{q2}) = \mathbf{S}_{q1} + \mathbf{S}_{q2} - \mathbf{S}_{q1} \circ \mathbf{S}_{q2} \\ \neg q : \quad & \mathcal{N}(\mathbf{S}_q) = 1 - \mathbf{S}_q \end{aligned}$$

More about fuzzy logic



t-norm ($\wedge$)		t-conorm ($\vee$)	Special Properties
minimum (Gödel)	$t(a,b) = \min(a,b)$	$s(a,b) = \max(a,b)$	idempotent
product	$t(a,b) = ab$	$s(a,b) = a+b-ab$	strict monotonicity
Łukasiewicz	$t(a,b) = \max(a+b-1,0)$	$s(a,b) = \min(a+b,1)$	nilpotent

- t-norm: conjunction (set intersection)
- t-conorm: disjunction (set union)
    - Defined by t-norm, negator: $c(x) = 1 - x$, and De Morgan's law

Well, why are they good?

- **Consistent with logical axioms!**
  - **An example of conjunction associativity**

$$\text{release(the Beatles, ?)} \land \text{compose(John Lennon, ?)} \land \text{compose(Paul McCartney, ?)}$$
$$\equiv \text{release(the Beatles, ?)} \land (\text{compose(John Lennon, ?)} \land \text{compose(Paul McCartney, ?)})$$

  - **Some previous methods (e.g. GQE) use averaging as logical operator conjunction**
    - **However, averaging is not associative!** $\frac{\frac{a+b}{2}+c}{2} \neq \frac{a+\frac{b+c}{2}}{2}$

Axiomatic systems of Boolean logic

- It is important to understand logic laws and take them into consideration when designing logical operators for query embedding models
  - Few efforts have been devoted into such theoretical analysis of query embedding models

To do that, we must understand how logical operations are defined

Axiomatic systems of Boolean logic

- Let $\mathcal{L}$ be the set of all the valid logic formulae under a logic system
- $\psi_1, \psi_2, \psi_3 \in \mathcal{L}$ represent logical formulae
- $I(\cdot)$ denotes the truth value of a logical formula

Axiomatic systems of Boolean logic

Semantics of Boolean logic is defined by:

- The interpretation $I\colon \mathcal{L} \to \{0,1\}$
    - The truth value of a logical formula
        - $\mathcal{L}$: the set of all the valid logic formulae

- The Modus Ponens inference rule / Logical Implication
    - From ψ_1 and $\psi_1 \to \psi_2$ infer ψ_2
    - This defines Logical implication:
        - $\boldsymbol{\psi_1 \to \psi_2}$ **holds if and only if** $\boldsymbol{I(\psi_2) \geq I(\psi_1)}$

- A set of axioms written in Hilbert-style deductive systems
    - Define other logic connectives via logic implication (→)

How is conjunction defined

- logical implication
  - $\psi_1 \to \psi_2$ holds if and only if $I(\psi_1) \le I(\psi_2)$

The following three axioms characterize $\wedge$:

$$\psi_1 \wedge \psi_2 \to \psi_1$$

$$\psi_1 \wedge \psi_2 \to \psi_2$$

Ensure that
$I(\psi_1 \wedge \psi_2) \le I(\psi_1)$
$I(\psi_1 \wedge \psi_2) \le I(\psi_2)$

$$(\psi_3 \to \psi_1) \to ((\psi_3 \to \psi_2) \to (\psi_3 \to \psi_1 \wedge \psi_2))$$

Ensure that $I(\psi_1 \wedge \psi_2) = 1$ if $I(\psi_1) = I(\psi_2) = 1$

They also imply commutativity and associativity of $\wedge$!

[Chvalovsky 2012]

From Boolean Logic to Fuzzy Logic

- $I(\psi_1)$ is in [0,1]
- Axioms preserve
  - All the operations in fuzzy logic will have the same results as Boolean logic, if the operations are applied to {0,1}
- Extra axioms to define logical operations for (0,1)

Connect to embedding model

Note:

$I(\mathit{Compose}(\text{John Lennon, Let It Be})) :=$
$\phi(\text{Compose(John Lennon, ?)}, \mathit{Let\ it\ Be})$

$e \in S_1 \land e \in S_2 \leftrightarrow e \in S_1 \land S_2$

- $I(\psi_1 \land \psi_2) \leq I(\psi_1)$

$$I(\mathit{Compose}(\text{John Lennon, Let It Be}) \land \mathit{Compose}(\mathit{Paul\ McCartney}, \text{Let It Be}))$$
$$\leq I(\mathit{Compose}(\text{John Lennon, Let It Be}))$$

Embedding model

Query

Entity

$$\underbrace{\phi(\text{Compose(John Lennon, ?)} \land \text{Compose(Paul McCartney, ?)}}_{\text{Query}}, \underbrace{\text{Let It Be}}_{\text{Entity}})$$
$$\leq \phi(\text{Compose(John Lennon, ?)}, \text{Let It Be})$$

Logical Laws and Model Properties

UCLA

Embedding model $\phi(q,e)$ estimates the probability that entity e answers query q

Axioms and derived logic laws in classical logic

Desired model property according to the logic law

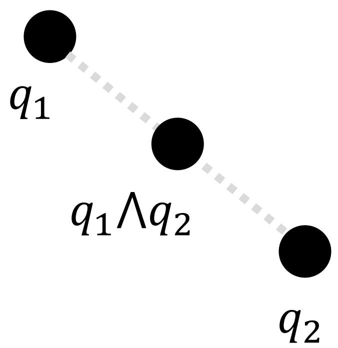
		Logic Law	Model Property
$\wedge$	I	Conjunction Elimination $\psi_1 \wedge \psi_2 \rightarrow \psi_1$ $\psi_1 \wedge \psi_2 \rightarrow \psi_2$	$\phi(q_1 \wedge q_2, e) \leq \phi(q_1, e)$ $\phi(q_1 \wedge q_2, e) \leq \phi(q_2, e)$
	II	Commutativity $\psi_1 \wedge \psi_2 \leftrightarrow \psi_2 \wedge \psi_1$	$\phi((q_1 \wedge q_2), e) = \phi((q_2 \wedge q_1), e)$
	III	Associativity $(\psi_1 \wedge \psi_2) \wedge \psi_3 \leftrightarrow$ $\psi_1 \wedge (\psi_2 \wedge \psi_3)$	$\phi((q_1 \wedge q_2) \wedge q_3, e)$ $= \phi(q_1 \wedge (q_2 \wedge q_3), e)$

		Logic Law	Model Property
$\wedge$	I	Conjunction Elimination $\psi_1 \wedge \psi_2 \rightarrow \psi_1$ $\psi_1 \wedge \psi_2 \rightarrow \psi_2$	 $\phi(q_1 \wedge q_2, e) \leq \phi(q_1, e)$ $\phi(q_1 \wedge q_2, e) \leq \phi(q_2, e)$
	II	Commutativity $\psi_1 \wedge \psi_2 \leftrightarrow \psi_2 \wedge \psi_1$	 $\phi((q_1 \wedge q_2), e) = \phi((q_2 \wedge q_1), e)$
	III	Associativity $(\psi_1 \wedge \psi_2) \wedge \psi_3 \leftrightarrow$ $\psi_1 \wedge (\psi_2 \wedge \psi_3)$	 $\phi((q_1 \wedge q_2) \wedge q_3, e)$ $= \phi(q_1 \wedge (q_2 \wedge q_3), e)$
$\vee$	IV	Disjunction Amplification $\psi_1 \rightarrow \psi_1 \vee \psi_2$ $\psi_1 \rightarrow \psi_1 \vee \psi_1$	 $\phi(q_1, e) \leq \phi(q_1 \vee q_2, e)$ $\phi(q_2, e) \leq \phi(q_1 \vee q_2, e)$
	V	Commutativity $\psi_1 \vee \psi_2 \leftrightarrow \psi_2 \vee \psi_1$	 $\phi((q_1 \vee q_2), e) = \phi((q_2 \vee q_1), e)$
	VI	Associativity $(\psi_1 \vee \psi_2) \vee \psi_3 \leftrightarrow$ $\psi_1 \vee (\psi_2 \vee \psi_3)$	 $\phi((q_1 \vee q_2) \vee q_3, e)$ $= \phi(q_1 \vee (q_2 \vee q_3), e)$
$\neg$	VII	Involution $\neg\neg\psi_1 \rightarrow \psi_1$	 $\phi(q, e) = \phi(\neg\neg q, e)$
	VIII	Non-Contradiction $\psi_1 \wedge \neg\psi_1 \rightarrow \overline{0}$	 $\phi(q, e) \uparrow \;\Rightarrow\; \phi(\neg q, e) \downarrow$

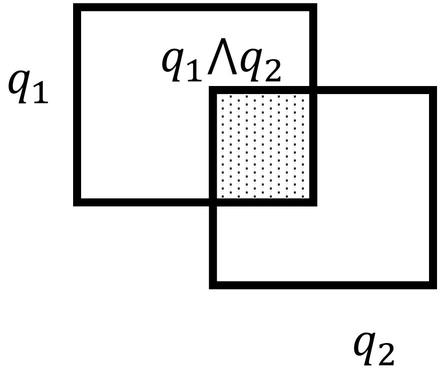
UCLA

Analysis of Previous Models' Capability of Preserving those Properties

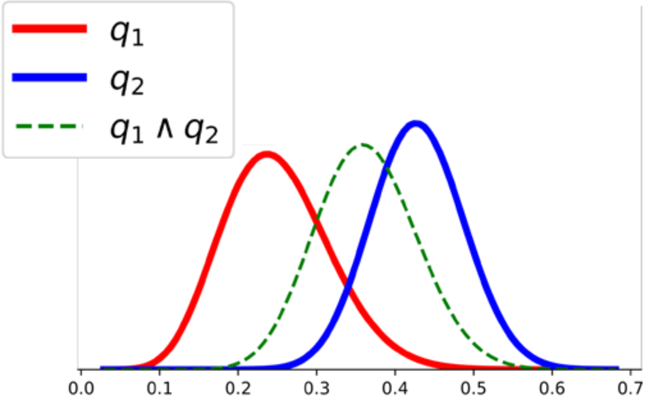
GQE



Query2Box



BetaE



	$\wedge$				$\vee$				$\neg$		
	Expr. (Closed)	*Com.*	*Asso.*	*Elim.*	*Expr.* (Closed)	*Com.*	*Asso.*	*Ampli.*	*Expr.* (Closed)	*Inv.*	*Non-Contra.*
GQE	✓(✓)	✓	✗	✗	✓(✗)	✓	✓	✓	✗	N/A	N/A
Query2Box	✓(✓)	✓	✓	✓	✓(✗)	✓	✓	✓	✗	N/A	N/A
BetaE	✓(✓)	✓	✗	✗	(i) DNF ✓(✗) (ii) DM ✓(✓)	✓ ✓	✓ ✓	✓ ✗	✓(✓)	✓	✗

None of previous models can satisfy all these properties

Analysis of Previous Models' Capability of Preserving those Properties

	$\wedge$				$\vee$				$\neg$		
	Expr. (Closed)	*Com.*	*Asso.*	*Elim.*	*Expr.* (Closed)	*Com.*	*Asso.*	*Ampli.*	*Expr.* (Closed)	*Inv.*	*Non-Contra.*
GQE	✓(✓)	✓	✗	✗	✓(✗)	✓	✓	✓	✗	N/A	N/A
Query2Box	✓(✓)	✓	✓	✓	✓(✗)	✓	✓	✓	✗	N/A	N/A
BetaE	✓(✓)	✓	✗	✗	(i) DNF ✓(✗) (ii) DM ✓(✓)	✓ ✓	✓ ✓	✓ ✗	✓(✓)	✓	✗
FuzzQE	✓(✓)	✓	✓	✓	✓(✓)	✓	✓	✓	✓(✓)	✓	✓

Our model can!

When complex training queries are not available

- Logical operators do not require learning any operator specific parameters

- It significantly outperforms previous query embedding models under the same training condition (using only KG edges)
  - Comparable to state-of-the-art query embedding models that are trained with extra complex query data

$$L = -\log \sigma(\phi(q,e) - \gamma) - \frac{1}{k}\sum_{i=1}^{k} \log \sigma(\gamma - \phi(q,e'))$$

Experimental Results

- **Train with only KG Edges (No Complex Queries)**

Table 6.8: **MRR results (%) of logical query embedding models that are trained with only link prediction.** This task tests the ability of the model to generalize to arbitrary complex logical queries, when no complex logical query data is available for training. Avg_{EPFO} and Avg_{Neg} denote the average MRR on EPFO $(\exists, \wedge, \vee)$ queries and queries containing negation respectively.

Model	Avg_{EPFO}	Avg_{Neg}	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
FB15k-237																
GQE	17.7	N/A	41.6	7.9	5.4	25.0	33.6	16.3	10.9	11.9	6.2	N/A	N/A	N/A	N/A	N/A
Query2Box	18.2	N/A	42.6	6.9	4.7	27.3	36.8	17.5	11.1	11.7	5.5	N/A	N/A	N/A	N/A	N/A
BetaE	15.8	0.5	37.7	5.6	4.4	23.3	34.5	15.1	7.8	9.5	4.5	0.1	1.1	0.8	0.1	0.2
FuzzQE	**21.8**	**6.6**	**44.0**	**10.8**	**8.6**	**32.3**	**41.4**	**22.7**	**15.1**	**13.5**	**8.7**	**7.7**	**9.5**	**7.0**	**4.1**	**4.7**
NELL995																
GQE	21.7	N/A	47.2	12.7	9.3	30.6	37.0	20.6	16.1	12.6	9.6	N/A	N/A	N/A	N/A	N/A
Query2Box	21.6	N/A	47.6	12.5	8.7	30.7	36.5	20.5	16.0	12.7	9.6	N/A	N/A	N/A	N/A	N/A
BetaE	19.0	0.4	53.1	6.0	3.9	32.0	37.7	15.8	8.5	10.1	3.5	0.1	1.4	0.1	0.1	0.1
FuzzQE	**27.1**	**7.3**	**57.6**	**17.2**	**13.3**	**38.2**	**41.5**	**27.0**	**19.4**	**16.9**	**12.7**	**9.1**	**8.3**	**8.9**	**4.4**	**5.6**

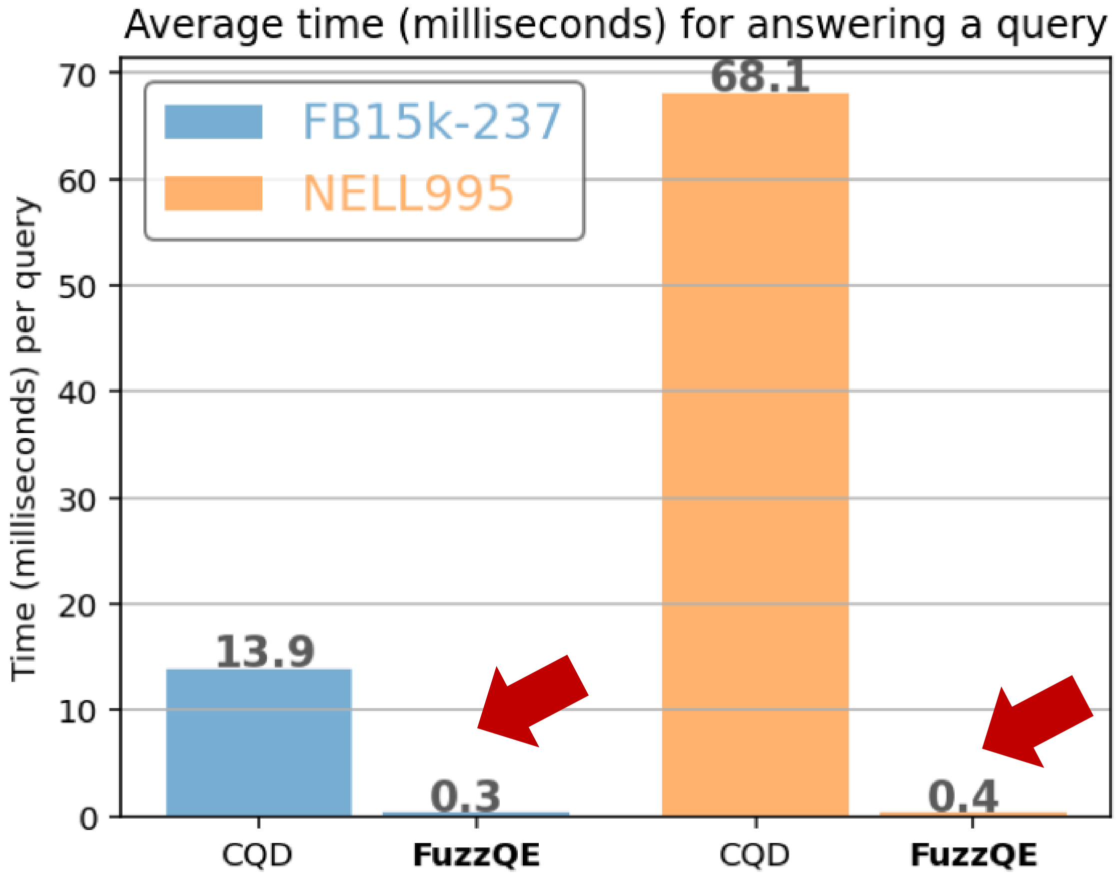
Experimental Results

- **Trained with additional complex queries**

Table 6.7: **MRR results (%) on answering FOL queries.** Report MRR results (%) on test FOL queries. Avg_{EPFO} and Avg_{Neg} denote the average MRR on EPFO queries (queries with $\exists$, $\wedge$, $\vee$ and without negation) and queries containing negation respectively. Results of GQE, Query2Box, and BetaE are taken from [81].

Type of Model	Model	Avg_{EPFO}	Avg_{Neg}	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
FB15k-237																	
Query Embedding	GQE	16.3	N/A	35.0	7.2	5.3	23.3	34.6	16.5	10.7	8.2	5.7	N/A	N/A	N/A	N/A	N/A
	Query2Box	20.1	N/A	40.6	9.4	6.8	29.5	42.3	21.2	12.6	11.3	7.6	N/A	N/A	N/A	N/A	N/A
	BetaE	20.9	5.5	39.0	10.9	10.0	28.8	42.5	22.4	12.6	12.4	9.7	5.1	7.9	7.4	3.5	3.4
	FuzzQE	**24.2**	**8.5**	42.2	**13.3**	**10.2**	**33.0**	**47.3**	**26.2**	**18.9**	**15.6**	**10.8**	**9.7**	**12.6**	**7.8**	**5.8**	**6.6**
Query Optimization	CQD	21.7	N/A	**46.3**	9.9	5.9	31.7	41.3	21.8	15.8	14.2	8.6	N/A	N/A	N/A	N/A	N/A
NELL995																	
Query Embedding	GQE	18.6	N/A	32.8	11.9	9.6	27.5	35.2	18.4	14.4	8.5	8.8	N/A	N/A	N/A	N/A	N/A
	Query2Box	22.9	N/A	42.2	14.0	11.2	33.3	44.5	22.4	16.8	11.3	10.3	N/A	N/A	N/A	N/A	N/A
	BetaE	24.6	5.9	53.0	13.0	11.4	37.6	47.5	24.1	14.3	12.2	8.5	5.1	7.8	10.0	3.1	3.5
	FuzzQE	**29.3**	**8.0**	58.1	**19.3**	**15.7**	39.8	**50.3**	**28.1**	**21.8**	**17.3**	**13.7**	**8.3**	**10.2**	**11.5**	**4.6**	**5.4**
Query Optimization	CQD	28.4	N/A	**60.0**	16.5	10.4	**40.4**	49.6	28.6	20.8	16.8	12.6	N/A	N/A	N/A	N/A	N/A

Compare with CQD Regarding Inference Time



- Average time (milliseconds) for answering an FOL query on a single NVIDIA GP102 TITAN Xp (12GB) GPU.
- FB15k-237 contains 14,505 entities.
- NELL995 contains 63,361 entities, roughly 4 times the number of FB15k-237.

Conclusion

- We propose a novel logical query embedding framework FuzzQE for answering complex logical queries on knowledge graphs.
- We present theoretical guidance for future work future research on embedding-based logical query answering models
- Extensive experiments show the promising capability of FuzzQE on answering logical queries on KGs.

Thank you!