A probabilistic approach to complex query answering

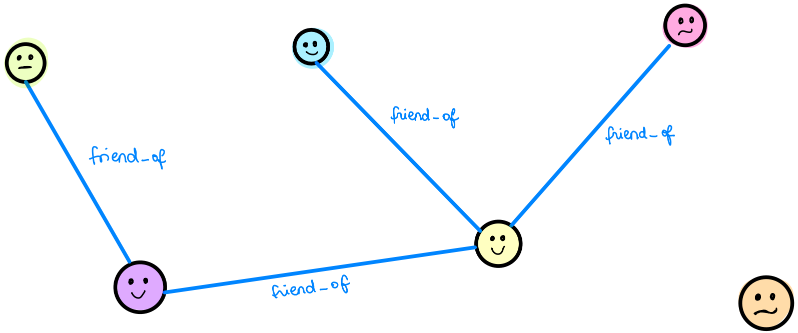
DBDBD2024

Tamara Cucumides
University of Antwerp

November 22, 2024

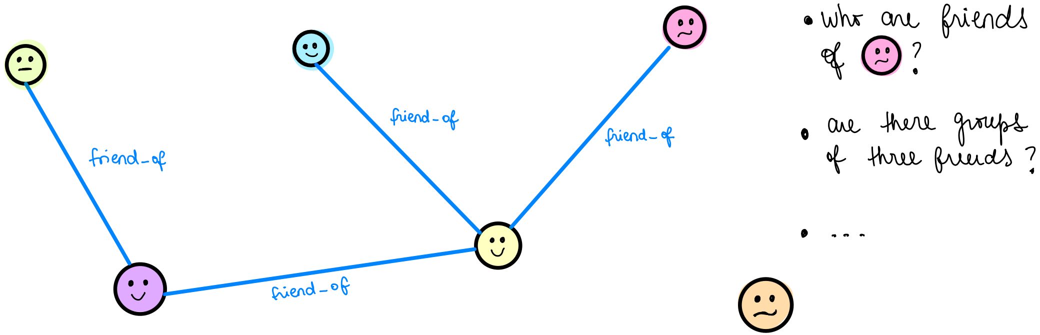
Motivation

Knowledge graphs



Motivation

Query answering on knowledge graphs

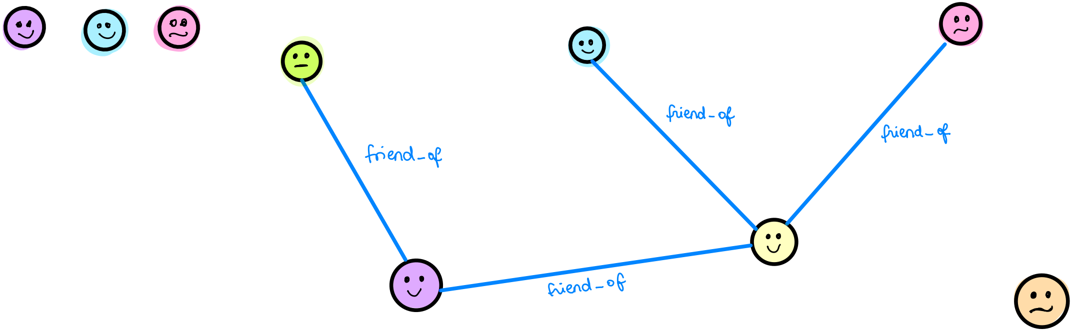


- Efficient algorithms has been developed to evaluate queries on knowledge graphs :)

Motivation

Query answering on **incomplete** knowledge graphs

Who are friends with ?



- most knowledge graphs are incomplete :(

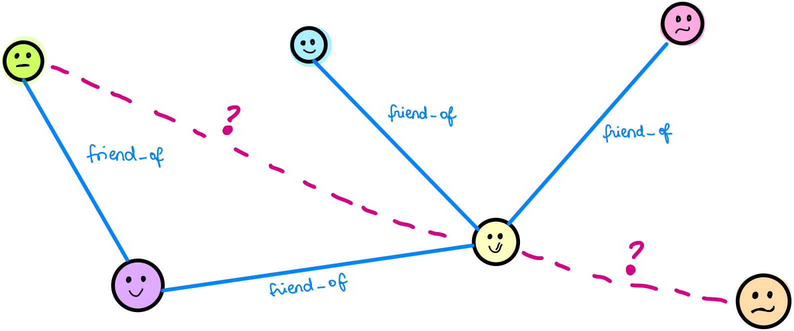
Motivation

Query answering on **incomplete** knowledge graphs

Who are friends with ?

what about

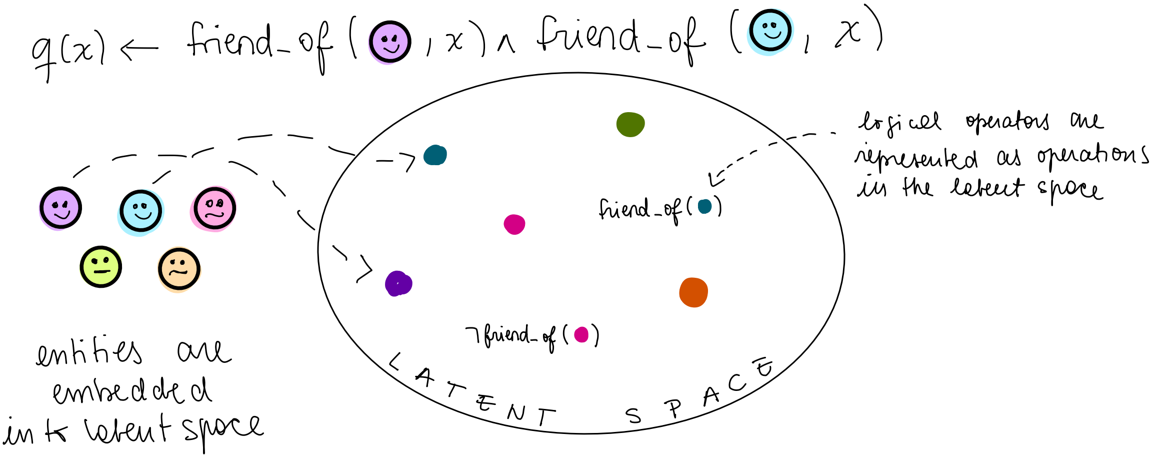
and ?



- most knowledge graphs are incomplete :(

Motivation

Current approaches come from the machine learning community



Motivation

Current approaches come from the machine learning community

CURRENT METHODS

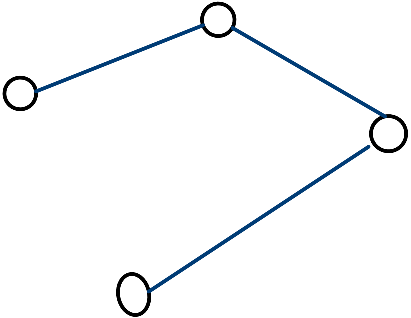
! • often support limited query types
• don't have clean semantics

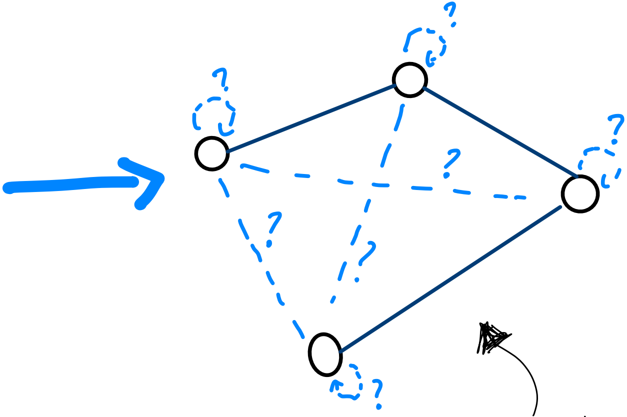
WE PROPOSE TO EXPLORE ANOTHER APPROACH...

Overview

Our approach

① "Complete" the graph





② Evaluate the query here!

Graph completion

What do we need?

1. An incomplete graph
2. Something to predict missing information
3. Space to store the completed graph

Graph completion

What do we need?

1. An incomplete graph
2. Something to predict missing information
3. Space to store the completed graph

Graph completion

To do the graph completion we use *link predictors*

Definition (Link predictor)

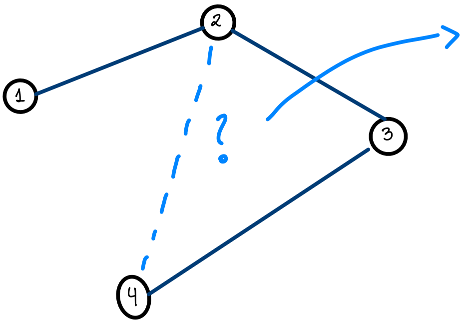
A link predictor maps facts to scores

$$\ell : V \times R \times V \longrightarrow [0, 1]$$

$$(u, R, v) \longmapsto \alpha$$

Graph completion

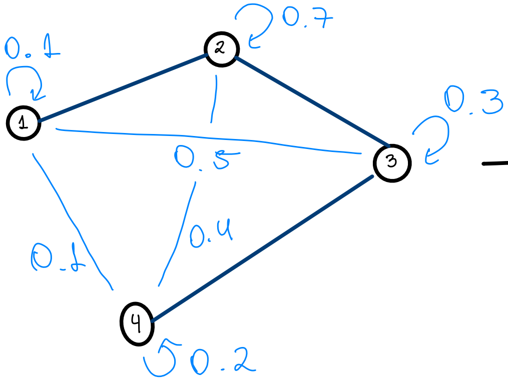
Using the link predictor, we *complete the graph*



For every missing fact.
(u, R, v)
We get a score that represents the likelihood that the fact exists in the knowledge graph.

Graph completion

Using the link predictor, we *complete the graph*



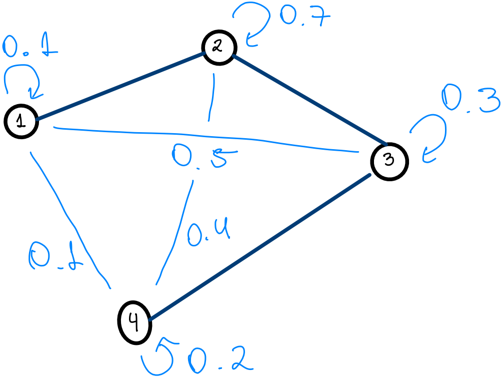
$$\begin{bmatrix} 0.1 & 1 & 0.5 & 0.1 \\ 1 & 0.7 & 1 & 0.4 \\ 0.5 & 1 & 0.3 & 1 \\ 0.1 & 0.4 & 1 & 0.2 \end{bmatrix}$$

Looks a lot like a probabilistic database...

Query evaluation

We draw techniques from probabilistic query evaluation, and use **possible worlds semantics**

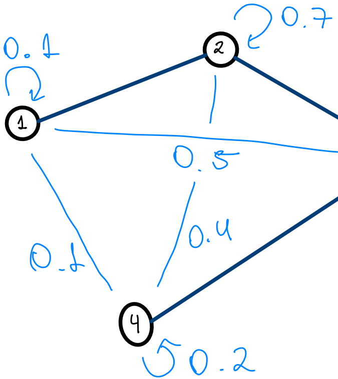
$$q(x) \leftarrow \exists z.\ R(u_1, z) \wedge R(z, x).$$



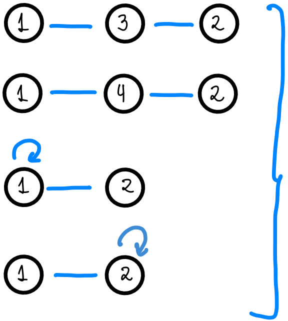
Query evaluation

We draw techniques from probabilistic query evaluation, and use **possible worlds semantics**

$$q(x) \leftarrow \exists z.\ R(u_1, z) \wedge R(z, x).$$



When is u_2 an answer?

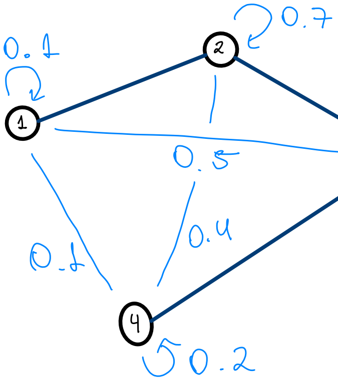


when one of this scenarios occur...

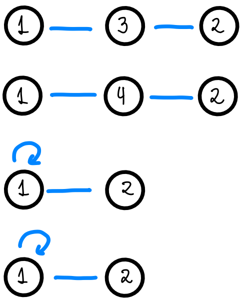
Query evaluation

We draw techniques from probabilistic query evaluation, and use **possible worlds semantics**

$$q(x) \leftarrow \exists z.\ R(u_1, z) \wedge R(z, x).$$



When is u_2 an answer?

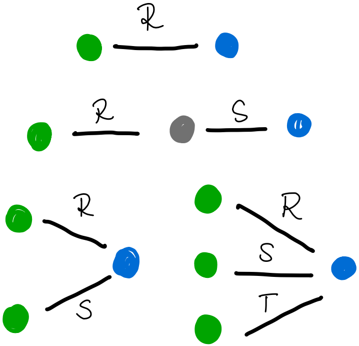


when one of this scenarios occur...

CALCULATING THIS IS IN GENERAL #P-hard.

Query evaluation

- FOR SOME QUERIES IT CAN BE DONE IN PTIME

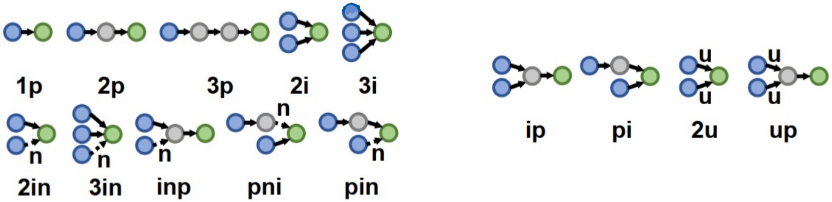


hierarchical queries

... and for the rest we compute approximations (dissociations).

Implementation and experimental setup

- As link predictor we use Neural Bellman-Ford Networks (NBFNets)
- Both the training of the link predictor and the graph completion process is done using GPUs
- We evaluate the model on the BetaE benchmark query set, plus extra cyclic queries



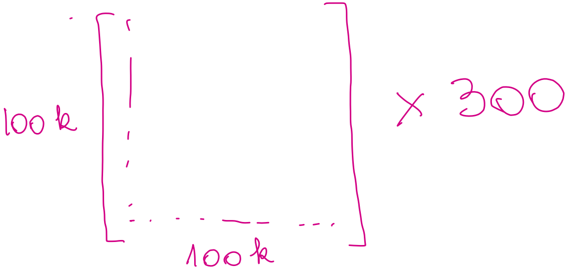
Practical challenges

One of the main reasons why this approach has been overlooked in previous work is because it comes with significant practical challenges

Efficiency

↳ materializing and storing the dense graphs is memory extensive.

⇝ BENCHMARK KG's have (hundred of) thousands entities, and hundreds different relations.



Efficiency

- USE OF SPARSE MATRICES.

Instead of storing dense matrices, we only keep scores above a threshold and use sparse matrices

$$\begin{bmatrix} 0.2 & 1.0 & 0.5 \\ 1.0 & 0.7 & 0.1 \\ 0.5 & 0.4 & 0.8 \end{bmatrix} \xrightarrow{t=0.3} \begin{bmatrix} - & 1.0 & 0.5 \\ 1.0 & 0.7 & - \\ 0.5 & 0.4 & 0.8 \end{bmatrix}$$

Efficiency

• USE OF SPARSE MATRICES.

Instead of storing dense matrices, we only keep scores above a threshold and use sparse matrices

$$\begin{bmatrix} 0.2 & 1.0 & 0.5 \\ 1.0 & 0.7 & 0.1 \\ 0.5 & 0.4 & 0.8 \end{bmatrix} \xrightarrow{t=0.3} \begin{bmatrix} - & 1.0 & 0.5 \\ 1.0 & 0.7 & - \\ 0.5 & 0.4 & 0.8 \end{bmatrix}$$

• MEMORY

- space can be reduced by 90% without compromising the performance of the model

• TIME

- evaluation becomes up to 10x faster for some query types.

Training

How do we train the link predictor?

① Traditional way.

- We feed the model true and false examples
- model is trained to give high scores to true examples and low scores to false ones.

EFFICIENT, BUT NOT IDEAL FOR QUERY ANS.

② Including queries

- We feed queries with true and false answers to the model
- model is trained to give high score to true answers and low scores to false ones

CAN ONLY BE DONE WITH SIMPLE QUERIES

- 40% performance improvement.

Evaluation metrics

Normally, the models for complex query answering are evaluated using ranking metrics

Evaluate $q(x)$ $\longrightarrow$

$u_1 : 0.7$
$u_2 : 0.9$
$u_3 : 0.2$
$u_4 : 0.5$
$u_5 : 0.8$

$\xrightarrow{\text{RANK}}$

u_2
u_5
u_1
u_4
u_3

Calculate mrr, hits@k against ground truth answers.

Evaluation metrics

Normally, the models for complex query answering are evaluated using ranking metrics

Evaluate $g(x)$ $\rightarrow$

$u_1 : 0.7$
$u_2 : 0.9$
$u_3 : 0.2$
$u_4 : 0.5$
$u_5 : 0.8$

$\xrightarrow{\text{RANK}}$

u_2
u_5
u_1
u_4
u_3

which ones are predicted as true answers?

BUT... this doesn't help us with the original task: QUERY ANSWERING.

Evaluation metrics

Normally, the models for complex query answering are evaluated using ranking metrics

→ BUT WE WANT TO CLASSIFY THE NODES TO EVALUATE THE QUERY

1 Choose of classification threshold

2 Training for ranking differs than training for classification

Open questions and future work

Although overlooked, the idea of *completing the graph* and further querying such completion could benefit from further exploration.

- can we train the link predictors using complex queries in an end-to-end schema?
- does improvement for graph completion translates to query answering?
- are there more efficient ways to evaluate queries in this scenario?

A probabilistic approach to complex query answering

DBDBD2024

Tamara Cucumides
University of Antwerp

November 22, 2024