Reasoning in Knowledge Graphs using Deep Learning

Joint work with H. Ren, W. Hamilton, R. Ying, J. You, M. Zitnik, D. Jurafsky

Jure Leskovec
Knowledge Graphs
Many Domains

- Wikidata
- The Economic Graph
- Company network
- Facebook network
Heterogeneous Networks

Knowledge Graphs

Heterogeneous Graphs

Multimodal Graphs
Knowledge Graphs

- Knowledge Graphs are heterogenous graphs
  - Multiple types of entities and relations exist
- Facts are represented as triples \((h, r, t)\)
  - (‘Alice’, ‘friend_with’, ‘Bob’)
  - (‘Paris’, ‘is_a’, ‘City’)
  - …
Traditional Tasks

Knowledge Graph Competition/Link Prediction

- Predict the missing head or tail for a given triple \((h, r, t)\)
- Example:

  Barack Obama \textcolor{red}{BornIn} United States

  Barack Obama \textcolor{red}{Nationality} American
Our goal: Reason over the knowledge graph using complex multi-hop queries

- Conjunctive queries: Subset of first-order logic with existential quantifier (\(\exists\)) and conjunction (\(\land\))

“Where did all Canadian citizens with Turing Award graduate?”

\[
q = V \cdot \exists V : \text{Win}(\text{Turing Award}, V) \land \text{Citizen}(\text{Canada}, V) \\
\land \text{Graduate}(V, V^?)
\]
“Where did Canadian citizens with Turing Award graduate?”

Each point corresponds to a set of entities
Why is it Hard?

- Heterogeneity: Lack of schema, or quite large schema (65K for DBpedia)
- Noise and incompleteness
- Uncertainty
- Massive size
- Fast query time

Relational Data (Structured) vs. Heterogeneous Graph Data (Semi-structured)
Why is it Hard?

**Key challenge:** Big graphs and queries can involve **noisy** and **unobserved** data!

Some links might be noisy or missing

**Problem:** Naïve link prediction and graph template matching are too expensive
**Our Idea: Query2Box**

Use representation learning to map a graph into a Euclidean space and learn to reason in that space.

Knowledge graph

Knowledge graph

**Logical query**

Reason in the embedding space

**Embedding space**

**ENC(u)**

**ENC(v)**

**encode nodes**

**Z_u**

**Z_v**
Semantic Embeddings

**TransE [Bordes et al., 2013]**

- Translation intuition:
  For a triple \((h, r, t)\): \(h + r \approx t\)

- **Q:** What is nationality of Obama?
Our Idea: **Query2Box**

**Idea:**

- 1) Embed nodes of the graph
- 2) For every logical operator learn a spatial operator

**So that:**

- 1) Take an arbitrary logical query. Decompose it into a set of logical operators ($\exists, \land, \lor$)
- 2) Apply a sequence of spatial operators to embed the query
- 3) Answers to the query are entities close to the embedding of the query
Our Idea: Query2Box

Idea:

1) Embed nodes of the graph

Key insight:

Represent query as a box.

Operations (union, intersection) are well defined over boxes.

3) Answers to the query are entities close to the embedding of the query
Embedding Queries

**Query2Box embedding:**

Embed queries with hyper-rectangles (boxes):

\[ q = (\text{Cen}(q), \text{Off}(q)) \].

[Probabilistic Embedding of Knowledge Graphs with Box Lattice Measures. Vilnis, et al., ACL 2018]
Embedding Queries

- Geometric Projection Operator
- Geometric Intersection Operator
Projection Operator

Geometric Projection Operator $\mathcal{P}$

- $\mathcal{P} : \text{Box} \times \text{Relation} \rightarrow \text{Box}$
Projection Operator: Example
Intersection Operator

Geometric Intersection Operator $I$

$\begin{align*}
I : \text{Box} \times \cdots \times \text{Box} & \rightarrow \text{Box} \\
& \text{The new center is a weighted average} \\
& \text{The new offset shrinks}
\end{align*}$
Intersection Operator: Example
How to Handle Disjunction

So far we can handle Conjunctive queries.

Can we learn a geometric disjunction operator?

- **Theorem (paraphrased):** For a KG with $M$ nodes, we need embedding dimension of $M$ to handle disjunction.
Disjunctive Normal Form

- Any query with AND and OR can be transformed into equivalent Disjunctive Normal Form (disjunction of conjunctive queries).
Disjunctions: Solution

Given an arbitrary AND-OR query

1) Transform it into an DNF
2) Answer each conjunctive query
3) Overall answer is the union of conjunctive query answers
Benefits of Query2Box

**Scalability and efficiency:**
- Any query can be reduced to a couple of matrix operations and a single k-nearest neighbor search

**Generality:**
- We can answer any query (even those we have never seen before)

**Robustness to noise:**
- Graph can contain missing and noisy relationships
“Where did Canadian citizens with Turing Award graduate?”

Dependency Graph

- Turing Award
- Win
- Citizen
- Graduate
- Canada

Example
“Where did Canadian citizens with Turing Award graduate?”

Dependancy Graph

Computation Graph

Embedding Process

Each point corresponds to a set of entities
“Where did Canadian citizens with Turing Award graduate?”

Dependency Graph

Computation Graph

Embedding Process

Each point corresponds to a set of entities
“Where did Canadian citizens with Turing Award graduate?”

**Dependency Graph**

**Computation Graph**

**Embedding Process**

Each point corresponds to a set of entities
“Where did Canadian citizens with Turing Award graduate?”

**Dependency Graph**

**Computation Graph**

**Embedding Process**

Each point corresponds to a set of entities
Query2Box: Model Training

Training examples: Queries on the graph

- **Positives:** Path with a known answer
- **Negatives:** Random nodes of the correct answer type
- **Goal:** Find embeddings and operators so that queries give correct answers
Experimental Setup

We essentially learn to “memorize” the answers to queries

- We embed entities so that our geometric operators give correct answers

Questions:

- Does our method generalize to new unseen queries?
- Does our method generalize to new query structures?
- Can method handle missing relations?
Experimental Setup

- **Training:**
  - Remove 10% of KG edges
  - Sample training queries and (non)answers
  - Train the model

- **Test set:**
  - Test queries/answers from the full graph
  - Ensure that the test queries are **not** directly answerable in the training graph
    - Every test query has at least one deleted edge
  - **Note:** Query template matching would have accuracy of random guessing
KG and Query Statistics

- **Freebase:** FB15K, FB15K-237

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Entities</th>
<th>Relations</th>
<th>Training Edges</th>
<th>Validation Edges</th>
<th>Test Edges</th>
<th>Total Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15k</td>
<td>14,951</td>
<td>1,345</td>
<td>483,142</td>
<td>50,000</td>
<td>59,071</td>
<td>592,213</td>
</tr>
<tr>
<td>FB15k-237</td>
<td>14,505</td>
<td>237</td>
<td>272,115</td>
<td>17,526</td>
<td>20,438</td>
<td>310,079</td>
</tr>
</tbody>
</table>

- **Queries:**

  - **Training Conjunctive Queries:**
    - 1p
    - 2p
    - 3p
    - 2i
    - 3i

  - **Unseen Conjunctive Queries:**
    - ip
    - pi

  - **Union Queries:**
    - 2u
    - up
Experimental Results

Observations:
- “Training” queries (1p-3p): +20%
- New conjunctive queries (2i,3i): +15%
- Disjunctive queries: +36%

<table>
<thead>
<tr>
<th>Method</th>
<th>Avg</th>
<th>1p</th>
<th>2p</th>
<th>3p</th>
<th>2i</th>
<th>3i</th>
<th>ip</th>
<th>pi</th>
<th>2u</th>
<th>up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q2B</td>
<td>0.268</td>
<td>0.467</td>
<td>0.24</td>
<td>0.186</td>
<td>0.324</td>
<td>0.453</td>
<td>0.108</td>
<td>0.205</td>
<td>0.239</td>
<td>0.193</td>
</tr>
<tr>
<td>Point-based embedding</td>
<td>0.228</td>
<td>0.402</td>
<td>0.213</td>
<td>0.155</td>
<td>0.292</td>
<td>0.406</td>
<td>0.083</td>
<td>0.170</td>
<td>0.169</td>
<td>0.163</td>
</tr>
<tr>
<td></td>
<td>0.230</td>
<td>0.405</td>
<td>0.213</td>
<td>0.153</td>
<td>0.298</td>
<td>0.411</td>
<td>0.085</td>
<td>0.182</td>
<td>0.167</td>
<td>0.160</td>
</tr>
</tbody>
</table>

Table 3: H@3 on test set for QUERY2BOX vs. GQE on FB15k-237.

<table>
<thead>
<tr>
<th>Method</th>
<th>Avg</th>
<th>1p</th>
<th>2p</th>
<th>3p</th>
<th>2i</th>
<th>3i</th>
<th>ip</th>
<th>pi</th>
<th>2u</th>
<th>up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q2B</td>
<td>0.484</td>
<td>0.786</td>
<td>0.413</td>
<td>0.303</td>
<td>0.593</td>
<td>0.712</td>
<td>0.211</td>
<td>0.397</td>
<td>0.608</td>
<td>0.330</td>
</tr>
<tr>
<td>Point-based embedding</td>
<td>0.386</td>
<td>0.636</td>
<td>0.345</td>
<td>0.248</td>
<td>0.515</td>
<td>0.624</td>
<td>0.151</td>
<td>0.31</td>
<td>0.376</td>
<td>0.273</td>
</tr>
<tr>
<td></td>
<td>0.384</td>
<td>0.63</td>
<td>0.346</td>
<td>0.250</td>
<td>0.515</td>
<td>0.611</td>
<td>0.153</td>
<td>0.32</td>
<td>0.362</td>
<td>0.271</td>
</tr>
</tbody>
</table>

Table 4: H@3 on test set for QUERY2BOX vs. GQE on FB15k.
FB15k: Embedding Space

“List male instrumentalists who play string instruments”
"List male instrumentalists who play string instruments"
"List male instrumentalists who play string instruments"

# of string instruments: 10

TPR: 100%
FPR: 0%
FB15k: Embedding Space

"List male instrumentalists who play string instruments"

# of instrumentalists: 472

TPR: 98.4%
FPR: 0.01%
“List male instrumentalists who play string instruments”
FB15k: Embedding Space

"List male instrumentalists who play string instruments"

# of men: 3555

TPR: 97.9%
FPR: 0.01%
FB15k: Embedding Space

“List male instrumentalists who play string instruments”

# of answers: 396

TPR: 99.4%
FPR: 0.01%
Query2Box: Embed the query as a box. Logical operations become spatial operations.

Composability of queries:
- Generalize well to unseen, extrapolated queries
- Explicitly training for composability is important

Instance vs. multi-hop generalization
Conclusion

- Box embeddings for answering logical queries on Knowledge graphs
- Handle union and intersection
- Generalize well to unseen, extrapolated queries
- Future work: Handle negation, other geometric model
References
