Ontology-Aware Partitioning for Knowledge Graph Identification

Jay Pujara, Hui Miao, Lise Getoor, William Cohen

Workshop on Automatic Knowledge Base Construction
10/27/2013
Knowledge Graph Ingredients: Entities

- Baltimore
- Orioles
- New York
- Giants
- San Francisco
- Maryland
- Baseball
- Football
- California
- New York
Knowledge Graph Ingredients: Labels

City
- Baltimore
- New York

Location
- San Francisco
- Maryland

State
- California
- New York

Sport
- Baseball
- Football

SportsTeam
- Orioles
- Giants
- Giants
Knowledge Graph Ingredients: Relations

City
- Baltimore
- New York
- San Francisco
- Maryland
- California
- New York

Location

State

Sport
- Football
- Baseball

SportsTeam
- Giants
- Giants
- Orioles
Knowledge Graph Ingredients: Ontology

City:
- Baltimore
- New York
- San Francisco

Location:
- Maryland
- California

State:
- New York

SportsTeam:
- Orioles
- Giants

Sport:
- Football
- Baseball
Ontological rules for Knowledge Graphs

Inverse:

\[ w_O : \text{INV}(R, S) \quad \land \quad \text{REL}(E_1, E_2, R) \Rightarrow \text{REL}(E_2, E_1, S) \]

Selectional Preference:

\[ w_O : \text{DOM}(R, L) \quad \land \quad \text{REL}(E_1, E_2, R) \Rightarrow \text{LBL}(E_1, L) \]
\[ w_O : \text{RNG}(R, L) \quad \land \quad \text{REL}(E_1, E_2, R) \Rightarrow \text{LBL}(E_2, L) \]

Subsumption:

\[ w_O : \text{SUB}(L, P) \quad \land \quad \text{LBL}(E, L) \Rightarrow \text{LBL}(E, P) \]
\[ w_O : \text{RSUB}(R, S) \quad \land \quad \text{REL}(E_1, E_2, R) \Rightarrow \text{REL}(E_1, E_2, S) \]

Mutual Exclusion:

\[ w_O : \text{MUT}(L_1, L_2) \quad \land \quad \text{LBL}(E, L_1) \Rightarrow \neg\text{LBL}(E, L_2) \]
\[ w_O : \text{RMUT}(R, S) \quad \land \quad \text{REL}(E_1, E_2, R) \Rightarrow \neg\text{REL}(E_1, E_2, S) \]

Adapted from Jiang et al., ICDM 2012
Knowledge Graph Identification (KGI)

- Joint inference over possible knowledge graphs
  - Resolves co-referent entities
  - Removes spurious labels and relations
  - Infers missing labels and relations
  - Uses many uncertain sources
  - Enforces ontological constraints

KGI: Under the Hood

- Define a probability distribution over knowledge graphs:

\[
P(G \mid D) = \frac{1}{Z} \exp \left[ - \sum_{r \in R} w_r \varphi_r(G) \right]
\]

- Hinge-Loss Markov Random Fields
  - Templated: easy to define with probabilistic soft logic
  - Continuous: atoms have continuous truth values in [0,1] range
  - Efficient: inference via convex optimization in O(|R|)

- Superior speed and quality for KGI (Pujara, ISWC13)

Problem: Knowledge Graphs are HUGE
Problem: Knowledge Graphs are HUGE
Solution: Partition the Knowledge Graph
Partitioning: advantages and drawbacks

- **Advantages**
  - Smaller problems
  - Parallel Inference
  - Speed / Quality Tradeoff

- **Drawbacks**
  - Partitioning large graph time-consuming
  - Key dependencies may be lost
  - New facts require re-partitioning
Key idea: Ontology-aware partitioning

- Partition the *ontology* graph, not the knowledge graph

- Induce a partitioning of the knowledge graph based on the ontology partition
Considerations: Ontology-aware Partitions

• Advantages:
  • Ontology is a smaller graph
  • Ontology coupled with dependencies
  • New facts can reuse partitions

• Disadvantages:
  • Insensitive to data distribution
  • All dependencies treated equally
Refinement: include data frequency

• Annotate each ontological element with its frequency

• Partition ontology with constraint of equal vertex weights
Refinement: weight edges by type

- Weight edges by their ontological importance
Datasets & Metrics

- Data from Never-ending Language Learner (NELL) from iteration 165
- Consists of over 1M extractions and a rich ontology
- Evaluation set from (Jiang, ICDM12) with 4.5K labeled extractions
- Report AUC-PR and running time, optimization terms of slowest partition

<table>
<thead>
<tr>
<th>Inputs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Candidate Labels</td>
<td>1.2M</td>
</tr>
<tr>
<td>Candidate Relations</td>
<td>100K</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Types</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique Labels</td>
<td>235</td>
</tr>
<tr>
<td>Unique Relations</td>
<td>221</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ontology</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dom</td>
<td>418</td>
</tr>
<tr>
<td>Rng</td>
<td>418</td>
</tr>
<tr>
<td>Inv</td>
<td>418</td>
</tr>
<tr>
<td>Sub</td>
<td>288</td>
</tr>
<tr>
<td>RSub</td>
<td>461</td>
</tr>
<tr>
<td>Mut</td>
<td>17.4K</td>
</tr>
<tr>
<td>RMut</td>
<td>48.5K</td>
</tr>
</tbody>
</table>
Experiments: Partitioning Approaches

Comparisons (6 partitions):

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NELL</td>
<td>Default promotion strategy, no KGI</td>
</tr>
<tr>
<td>KGI</td>
<td>No partitioning, full knowledge graph model</td>
</tr>
<tr>
<td>baseline</td>
<td>KGI, Randomly assign extractions to partition</td>
</tr>
<tr>
<td>Ontology</td>
<td>KGI, Edge min-cut of ontology graph</td>
</tr>
<tr>
<td>O+Vertex</td>
<td>KGI, Weight ontology vertices by frequency</td>
</tr>
<tr>
<td>O+V+Edge</td>
<td>KGI, Weight ontology edges by inv. frequency</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>AUPRC</th>
<th>Running Time</th>
<th>Opt. Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>NELL</td>
<td>0.765</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>KGI</td>
<td>0.794</td>
<td>97</td>
<td>10.9M</td>
</tr>
<tr>
<td>baseline</td>
<td>0.780</td>
<td>31</td>
<td>3.0M</td>
</tr>
<tr>
<td>Ontology</td>
<td>0.788</td>
<td>42</td>
<td>4.2M</td>
</tr>
<tr>
<td>O+Vertex</td>
<td>0.791</td>
<td>31</td>
<td>3.7M</td>
</tr>
<tr>
<td>O+V+Edge</td>
<td>0.790</td>
<td>31</td>
<td>3.7M</td>
</tr>
</tbody>
</table>
Experiments: Scalability (Ont+Vertex)

Partitions vs. Performance

Running Time (minutes)

Number of Partitions

Area Under Precision-Recall Curve

0.794

0.791

0.791

0.790

0.788

0.5

0.55

0.6

0.65

0.7

0.75

0.8

0.85

0.9

0.95

1

1

2

3

6

12

24

48

0.5

0.55

0.6

0.65

0.7

0.75

0.8

0.85

0.9

0.95

1
Conclusion

- Knowledge Graph Identification: a powerful technique for constructing consistent knowledge graphs...
- Scalability is still a concern for real-world applications
- Partitioning can address scalability concerns
  - Ontology-aware partitioning partitions the *ontology* instead of the knowledge graph
  - Including data distribution information balances partitions
- Results on the NELL dataset show this strategy reduces running time from 97 minutes to 12 minutes without significant AUC degradation

Code Available on GitHub: https://github.com/linqs/KnowledgeGraphIdentification