Knowledge Graph Completion
Introduction and motivation

We have our ‘constructed’ knowledge graph, now what?
Introduction and motivation

**Problem 1:** Wrong/missing triples
Introduction and motivation

Problem 2: Many nodes refer to the same underlying entity
For Web extractions, noise is inevitable

- Thousands of web domains
- Many page formats
- Distracting & irrelevant content
- Purposeful obfuscation
- Poor grammar & spelling
- Tables

To reach its potential, a constructed KG must be completed or identified
Noise Analysis

- Extractors found to offer a collective tradeoff between multiple dimensions
  - Noise is rarely ‘random’!

<table>
<thead>
<tr>
<th>Easy to define</th>
<th>Glossary</th>
<th>Regex</th>
<th>Landmark</th>
<th>CRF</th>
<th>NER</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Site coverage</th>
<th>Glossary</th>
<th>Regex</th>
<th>Landmark</th>
<th>CRF</th>
<th>NER</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>All</td>
<td>Short Tail</td>
<td>All</td>
<td>All</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Precision</th>
<th>Glossary</th>
<th>Regex</th>
<th>Landmark</th>
<th>CRF</th>
<th>NER</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-3</td>
<td>3-4</td>
<td>4</td>
<td>2-3</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Recall</th>
<th>Glossary</th>
<th>Regex</th>
<th>Landmark</th>
<th>CRF</th>
<th>NER</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
ENTITY RESOLUTION
Definitions and alternate names

- **Common sense:**
  - Which entities refer to the same thing?

- **Slightly more formal:**
  - Which mentions (aka records, instances, nodes, surface strings…) refer to the same underlying entity?

- **Rigorous mathematical/logical definition**
  - Doesn’t exist, or unknown! Just like other hard AI problems...

- **Why try to solve the problem aka why is it a problem?**
Applications: A Web of Linked ‘Data’
Applications: Schema.org

- **Schema.org** is an RDF ontology from which triples (with Web-dereferencable URIs) can be embedded in HTML pages.
Applications: Google Knowledge Graph

https://developers.google.com/knowledge-graph/
SUB-COMMUNITIES
Entity Linking/Canonicalization

- Name of an entity (such as a city or location) not enough to resolve ambiguity
- Use Geonames knowledge base to canonicalize entity using machine learning and text features

- Berlin, California, the former name of Genevra, California
- Berlin, Connecticut
  - Berlin (Amtrak station), rail station in Berlin, Connecticut
- Berlin, Georgia
- Berlin, Illinois
- Berlin, Indiana, extinct town
- Berlin, Kansas
- Berlin, Kentucky
- Berlin, Maryland
- Berlin, Massachusetts
- Berlin, Michigan (disambiguation)
- Berlin, Nebraska, a former name of Otoe, Nebraska
Co-reference Resolution

coreference (discourse)

Wikinews interviews President of the International Brotherhood of Magicians Wednesday, October 9, 2013 October is National Magic Month in the United States. Wikinews spoke with William Evans, president of the International Brotherhood of Magicians, about the current state of magic and what its future looks like in the world of entertainment. For how long have you been involved in performing / studying magic? Over 50 years. I am 61 now so I really started when I was about 10
Entity Resolution (what we’ll be covering)

- Itself has many sub-communities and approaches
- Because of flexible representations (compared to databases or strict models like OWL), KG-ER systems tend to be community-agnostic
STANDARD ER ARCHITECTURE
Entity Resolution is fundamentally non-linear

- Theoretically quadratic in the number of nodes, even if ‘resolution rule’ was known
- In practice, number of ‘duplicates’ tends to grow linearly, and duplicates overlap in non-trivial ways
- How to devise efficient algorithms?

50 years of research has agreed on a two-step solutions
Blocking

• Key idea is to use a cheap heuristic that efficiently clusters approximately similar entities into (possibly overlapping) blocks.

Apply blocking key e.g. Tokens(LastName)

‘Exhaustive’ set: $10 \choose 2 = 45$ pairs

Generate candidate set (12 pairs), apply similarity function on each pair
Aside: some blocks have skewed size...

- Property of real-world data (zipf distribution, power laws...)
- How to address data skew?
  - Apply blocking methods with guarantees
  - May lose some recall in the process

Example

**Sorted Neighborhood** aka merge-purge:
- Use blocking key as ‘sorting’ key
- Slide a window of constant size \((w)\) over sorted nodes
- Only pairs of nodes within window are paired, added to candidate set

<table>
<thead>
<tr>
<th>ID</th>
<th>First Name</th>
<th>Last Name</th>
<th>Zip</th>
<th>BKV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cathy</td>
<td>Ransom</td>
<td>77111</td>
<td>CR7</td>
</tr>
<tr>
<td>2</td>
<td>Catherine</td>
<td>Ridley</td>
<td>77093</td>
<td>CR7</td>
</tr>
<tr>
<td>3</td>
<td>Cathy</td>
<td>Ridley</td>
<td>77093</td>
<td>CR7</td>
</tr>
<tr>
<td>4</td>
<td>John</td>
<td>Rogers</td>
<td>78751</td>
<td>JR7</td>
</tr>
<tr>
<td>5</td>
<td>J.</td>
<td>Rogers</td>
<td>78732</td>
<td>JR7</td>
</tr>
<tr>
<td>6</td>
<td>John</td>
<td>Ridley</td>
<td>77093</td>
<td>JR7</td>
</tr>
<tr>
<td>7</td>
<td>John</td>
<td>Ridley Sr.</td>
<td>77093</td>
<td>JRS7</td>
</tr>
</tbody>
</table>

Final Candidate Set \((w = 3)\):
\[
\{(1,2), (2,3), (1,3), (2,4), (3,4), (3,5), (4,5), (4,6), (5,6), (5,7), (6,7)\}

Other methods: block purging, canopies...
Similarity/link specification

• Over 50 years of research on what makes for a good ‘similarity’ function
• Current approach: apply ‘typical’ machine learning workflow to candidate set
• Important to remember that features are extracted from ‘mention pairs’...leads to non-trivial alignment issues
  – Some form of schema-matching almost always attempted in practical systems
  – Some (but not much) work on so-called schema-free similarity

General

Semantic Web
Aside: why schema matching?
Feature engineering

Open question: how much can representation learning contribute to Entity Resolution?
Similarity: putting it together

- ML model can be supervised, semi-supervised or unsupervised

Schema alignment /extract useful information sets

<table>
<thead>
<tr>
<th>Candidate set</th>
<th>Machine Learning (ML) model</th>
</tr>
</thead>
</table>

Probability that pair is duplicate

- Fenix at the Argyle
  - 8358 Sunset Blvd.
- Hollywood
  - American
- 8358 Sunset Blvd.
  - W. Hollywood
  - French (new)

StringsSim:
- [0.1, 0.6, ..., 0.9, ..., 0]
OUTPUT REPRESENTATION AND HANDLING
From links to clusters

• For perfect links, transitive closure/connected components works

• With imperfect links, effect can be severe
  – One weak link is all it takes to form a giant component
  – Not uncommon in the real world

• More robust clustering methods have to be applied
  – Community detection literature
  – Spectral clustering
  – Many more!

• Some recent work has proposed to explore ER as a micro-clustering problem
From (possibly noisy) clusters to…???

• Surprisingly under-studied problem!
• Should the entities be fused into a single entity? How?
  – Entity linking has a conceptually elegant solution to this problem…
  – …but how to deal with NIL clusters?
• Semantic Web approach
  – Represent individual links as KG triples and leave it at that
  – Entity Name Systems for advanced search/reasoning
BEYOND ENTITY RESOLUTION
By itself, generic ER is unlikely to be enough to sufficiently boost KG quality

• Other things explored in the literature:

  • Domain knowledge
    – Collective ER methods have tried to exploit these systematically

  • Multi-type Entity Resolution
    – Extremely useful for knowledge graphs, lots more work to be done

  • Entity Resolution+Ontologies+IE Confidences:
    – Probabilistic Graphical Models like Probabilistic Soft Logic

  • Knowledge graph embeddings
    – Useful for link prediction and triples classification
    – Recall the Microsoft-founded_in-Seattle example earlier
Knowledge graph embeddings/representation learning

- Useful for link prediction/missing relationships/triples classification
- Not clear if it is really better than PSL on noisy KGs
- Not clear how to combine KGEs with domain engineering

<table>
<thead>
<tr>
<th>Model</th>
<th>#Parameters</th>
<th># Operations (Time complexity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unstructured (Bordes et al. 2012; 2014)</td>
<td>$O(N_e m)$</td>
<td>$O(N_t)$</td>
</tr>
<tr>
<td>SE (Bordes et al. 2011)</td>
<td>$O(N_e m + 2N_r n^2)(m = n)$</td>
<td>$O(2m^2 N_t)$</td>
</tr>
<tr>
<td>SME(linear) (Bordes et al. 2012; 2014)</td>
<td>$O(N_e m + N_r n + 4mk + 4k)(m = n)$</td>
<td>$O(4m k N_t)$</td>
</tr>
<tr>
<td>SME (bilinear) (Bordes et al. 2012; 2014)</td>
<td>$O(N_e m + N_r n + 4mks + 4k)(m = n)$</td>
<td>$O(4mks N_t)$</td>
</tr>
<tr>
<td>LFM (Jenatton et al. 2012; Sutskever et al. 2009)</td>
<td>$O(N_e m + N_r n^2)(m = n)$</td>
<td>$O((m^2 + m) N_t)$</td>
</tr>
<tr>
<td>SLM (Socher et al. 2013)</td>
<td>$O(N_e m + N_r (2k + 2nk))(m = n)$</td>
<td>$O((2mk + k) N_t)$</td>
</tr>
<tr>
<td>NTN (Socher et al. 2013)</td>
<td>$O(N_e m + N_r (n^2s + 2ns + 2s))(m = n)$</td>
<td>$O(((m^2 + m)s + 2mk + k) N_t)$</td>
</tr>
<tr>
<td>TransE (Bordes et al. 2013)</td>
<td>$O(N_e m + N_r n)(m = n)$</td>
<td>$O(N_t)$</td>
</tr>
<tr>
<td>TransH (Wang et al. 2014)</td>
<td>$O(N_e m + 2N_r n)(m = n)$</td>
<td>$O(2m N_t)$</td>
</tr>
<tr>
<td>TransR (Lin et al. 2015)</td>
<td>$O(N_e m + N_r (m + 1)n)$</td>
<td>$O(2m n N_t)$</td>
</tr>
<tr>
<td>CTransR (Lin et al. 2015)</td>
<td>$O(N_e m + N_r (m + d)n)$</td>
<td>$O(2m n N_t)$</td>
</tr>
</tbody>
</table>
Entity Resolution (ER) is a hard problem for machines, may be AI complete
   – It’s ‘easy’ for us because we’re so good at it
   – Not clear what will achieve the next breakthrough in ER

Essential to attempt a solution if KGs are semi-automatically constructed from Web data
   – Quality doesn’t have to be perfect, as we showed earlier with KG search

Wealth of solutions but can be broken down into standard components
   – Blocking, to make ER efficient
   – Similarity, to make ER automaticadaptive

Many open questions, especially in relation to new ML models
More broadly, lots of opportunities for KG completion
• Allemang, D., & Hendler, J. (2011). *Semantic web for the working ontologist: effective model*


