Knowledge Graph Completion

Mayank Kejriwal (USC/ISI)
What is knowledge graph completion?

• An ‘intelligent’ way of doing data cleaning
  • Deduplicating entity nodes (entity resolution)
  • Collective reasoning (probabilistic soft logic)
  • Link prediction
  • Dealing with missing values
  • Anything that improves an existing knowledge graph!

• Also known as knowledge base identification
Some solutions we’ll cover today

• Entity Resolution (ER)
• Probabilistic Soft Logic (PSL)
• Knowledge Graph Embeddings (KGEs), with applications
Entity Resolution (ER)
Entity Resolution (ER)

- The **algorithmic** problem of grouping entities referring to the *same* underlying entity
Aside: Resolving Entity Resolution

- Itself has many alternate names in the research community!

*Many thanks to Lise Getoor*
ER is less constrained for graphs than tables (why?)
KG nodes are multi-type
Two KGs may be published under different ontologies.
How to do ER?

• Popular methods use some form of machine learning; see surveys by Kopcke and Rahm (2010), Elmagarmid et al. (2007), Christophides et al. (2015)

**Probabilistic Matching Methods**
- Marlin (SVM based)
- Bilenko and Mooney (2003)

**Supervised, Semi-supervised**

**Active Learning**

**Rule Based**

**Distance Based**

**Unsupervised**
- EM
- Winkler (1993)
- Hierarchical Graphical Models
- SVM
- Christen (2008)
With graph representation

• Can **propagate** similarity decisions Melnik, Garcia-Molina and Rahm (2002)
  • More expensive but better performance

• Can be **generic** or use **domain knowledge** e.g., citation/bibliography domain Bhattacharya and Getoor (2006, 2007)
Example (co-authorship)

- Bhattacharya and Getoor (2006, 2007)
Example (co-authorship)

- Bhattacharya and Getoor (2006, 2007)
Example (co-authorship)

- Bhattacharya and Getoor (2006, 2007)
Example (co-authorship)

- Bhattacharya and Getoor (2006, 2007)
Feature functions - I

• First line of attack is *string matching*

Available Packages: *SecondString, FEBRL, Whirl...*
Learnable string similarity

• Example: adaptive edit distance

Sets of equivalent string pairs (e.g., <Suite 1001, Ste. 1001>)

Learned parameters

Bilenko and Mooney (2003)
After training...

• Apply classifier i.e. link specification function to every pair of nodes? Quadratic complexity!

\[ O(|V|^2) \]

Applications of similarity function

Linked mentions
More formally

- Input: Two graphs $G$ and $H$ with $|V|$ nodes each, pairwise Link Specification Function (LSF) $L$
- Naïve algorithm: Apply $L$ on $|V| \times |V|$ node pairs, output pairs flagged (possibly probabilistically) by function

Complexity is quadratic: $O(T(L)|V|^2)$

How do we reduce the number of applications of $L$?
Blocking trick

• Like a configurable inverted index function
What is a good blocking key?

• Achieves high recall
• Achieves high reduction
• Good survey on blocking: Christen (2012)
How do we learn a good blocking key?

• Key idea in existing work is to learn a DNF rule with indexing functions as atoms

CharTriGrams(Last_Name) U (Numbers(Address) X Last4Chars(SSN))

Putting it together

Training set of duplicates/non-duplicates

- Learn blocking key
- Blocking key
- Execute blocking
- RDF dataset 1
- RDF dataset 2

- Learn Similarity function
- Trained Classifier
- Candidate set
- Execute similarity
- :sameAs links
Post-processing step: soft transitive closure

• How do we combine :sameAs links into groups of unique entities?
  • Naïve transitive closure might not work due to noise!
• Clustering and ‘soft transitive closure’ algorithms could be applied
• Not as well-studied for ER
  • Has unique properties! ER is a micro-clustering problem
  • How to incorporate collective reasoning (better-studied)?
  • Efficiency!
ER packages

- Several are available, but some may need tuning to work for RDF
  - FEBRL was designed for biomedical record linkage (Christen, 2008)
  - Dedupe [https://github.com/dedupeio/dedupe](https://github.com/dedupeio/dedupe)
  - LIMES, Silk mostly designed for RDF data (Ngonga Ngomo and Auer, 2008; Isele et al. 2010)
Not all attributes are equal

• Phones/emails important in domains like organizations
  • (names are unreliable)

• Names can be important in certain domains
  • (nothing special about phones)

• How do we use this knowledge?
Domain knowledge

• Especially important for unusual domains but how do we express and use it?
  • Use rules? Too brittle, don’t always work!

• Use machine learning? Training data hard to come by, how to encode rule-based intuitions?
Summary

• **Entity Resolution** is the first line of attack for the knowledge graph completion problem

• The problem is usually framed in terms of two steps: **blocking** and **similarity** (or link specification)
  • Blocking is used for reducing exhaustive pairwise **complexity**
  • Similarity determines what makes two things the same
  • Both can use **machine learning**!

• Many open research sub-problems, especially in SW
Probabilistic Soft Logic (PSL)

Many thanks to Jay Pujara for his inputs/slides
Collective Reasoning over Noisy Extractions

- Noise in extractions is not random
- Jointly reason over facts and extractions to converge to the most probable extractions
- Use a combination of logic, semantics and machine learning for best performance (but how?)
Internet (noisy) Extraction Graph

Large-scale IE

Knowledge Graph

= Large-scale IE

Joint Reasoning
Extraction Graph

Uncertain Extractions:
.5: Lbl(Kyrgyzstan, bird)
.7: Lbl(Kyrgyzstan, country)
.9: Lbl(Kyrgyz Republic, country)
.8: Rel(Kyrgyz Republic, Bishkek, hasCapital)
**Uncertain Extractions:**
- 0.5: Lbl(Kyrgyzstan, bird)
- 0.7: Lbl(Kyrgyzstan, country)
- 0.9: Lbl(Kyrgyz Republic, country)
- 0.8: Rel(Kyrgyz Republic, Bishkek, hasCapital)

**Ontology:**
- Dom(hasCapital, country)
- Mut(country, bird)

**Entity Resolution:**
- SameEnt(Kyrgyz Republic, Kyrgyzstan)

**Extraction Graph + Ontology + ER**
Extraction Graph+Ontology + ER+PSL

Uncertain Extractions:
.5: Lbl(Kyrgyzstan, bird)
.7: Lbl(Kyrgyzstan, country)
.9: Lbl(Kyrgyz Republic, country)
.8: Rel(Kyrgyz Republic, Bishkek, hasCapital)

Ontology:
Dom(hasCapital, country)
Mut(country, bird)

Entity Resolution:
SameEnt(Kyrgyz Republic, Kyrgyzstan)

After Knowledge Graph Identification

(Annotated) Extraction Graph
Probabilistic Soft Logic (PSL)

- Templating language for hinge-loss MRFs, very scalable!
- Model specified as a **collection of logical formulas**

\[
\text{SAMEENT}(E_1, E_2) \tilde{\wedge} \text{LBL}(E_1, L) \Rightarrow \text{LBL}(E_2, L)
\]

- Uses **soft-logic** formulation
  - Truth values of atoms relaxed to [0,1] interval
  - Truth values of formulas derived from Lukasiewicz t-norm
Technical Background: PSL Rules to Distributions

• Rules are *grounded* by substituting literals into formulas

\[ w_{EL} : \text{SAMEENT(Kyrgyzstan, Kyrgyz Republic) } \sim \text{LBL(Kyrgyzstan, country)} \Rightarrow \text{LBL(Kyrgyz Republic, country)} \]

• Each ground rule has a weighted *distance to satisfaction* derived from the formula’s truth value

\[ P(G \mid E) = \frac{1}{Z} \exp \left( -w_r \sum_r (G) \right) \]

• The PSL program can be interpreted as a joint probability distribution over all variables in knowledge graph, conditioned on the extractions
Finding the best knowledge graph

- **Most probable explanation** (MPE) inference solves $\max_G P(G)$ to find the best KG

- In PSL, inference solved by **convex** optimization

- **Efficient:** running time scales with $O(|R|)$
PSL Rules: Uncertain Extractions

Weight for source $T$ (relations)

$w_{CR-T} : \text{CANDREL}_T(E_1, E_2, R)$

Predicate representing uncertain relation extraction from extractor $T$

$\Rightarrow \text{REL}(E_1, E_2, R)$

Relation in Knowledge Graph

Weight for source $T$ (labels)

$w_{CL-T} : \text{CANDLBL}_T(E, L)$

Predicate representing uncertain label extraction from extractor $T$

$\Rightarrow \text{LBL}(E, L)$

Label in Knowledge Graph
PSL Rules: Entity Resolution

\[ w_{EL} : \text{SAMEENT}(E_1, E_2) \land LBL(E_1, L) \Rightarrow LBL(E_2, L) \]
\[ w_{ER} : \text{SAMEENT}(E_1, E_2) \land \text{REL}(E_1, E, R) \Rightarrow \text{REL}(E_2, E, R) \]
\[ w_{ER} : \text{SAMEENT}(E_1, E_2) \land \text{REL}(E, E_1, R) \Rightarrow \text{REL}(E, E_2, R) \]

ER predicate captures confidence that entities are co-referent

- Rules require co-referent entities to have the same labels and relations
- Creates an *equivalence class* of co-referent entities
PSL Rules: Ontology

Inverse:

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

Selectional Preference:

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

Subsumption:

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

Mutual Exclusion:

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

Adapted from Jiang et al., ICDM 2012
Evaluated extensively: case study on NELL

**Task:** Compute a full knowledge graph from uncertain extractions

**Comparisons:**

**NELL**  
NELL’s strategy: ensure ontological consistency with existing KB

**PSL-KGI**  
Apply full Knowledge Graph Identification model

**Running Time:** Inference completes in 130 minutes, producing 4.3M facts

<table>
<thead>
<tr>
<th></th>
<th>AUC</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NELL</td>
<td>0.765</td>
<td>0.801</td>
<td>0.477</td>
<td>0.634</td>
</tr>
<tr>
<td>PSL-KGI</td>
<td>0.892</td>
<td>0.826</td>
<td>0.871</td>
<td>0.848</td>
</tr>
</tbody>
</table>
Summary

• Probabilistic Soft Logic (PSL) is a powerful framework for producing knowledge graphs from noisy IE and ER outputs

• PSL can be used to enforce global ontological constraints and capture uncertainty in the model

• The model is scalable i.e. it infers complete knowledge graphs for datasets with millions of extractions

Very well-documented and maintained: code, tutorials and publications openly available:

https://github.com/lingis/psl
Knowledge Graph Embeddings (KGEs)
Low-dimensional vector spaces

• Very popular for documents, graphs, words...
Some more intuition

• Embeddings are not a ‘new’ invention...topic models are an early example still widely used!
Knowledge graph embeddings

• Many ways to model the problem: entities are usually vectors, relations could be vectors or matrices
### Objective/loss/energy functions

- What is an ‘optimal’ vector/matrix for an entity or relation?

<table>
<thead>
<tr>
<th>Model</th>
<th>Score function $f_r(h, t)$</th>
<th># Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE (Bordes et al. 2013b)</td>
<td>$|h + r - t|_{\ell_1/2}$, $r \in \mathbb{R}^k$</td>
<td>$O(n_e k + n_r k)$</td>
</tr>
<tr>
<td>Unstructured (Bordes et al. 2012)</td>
<td>$|h - t|^2$</td>
<td>$O(n_e k)$</td>
</tr>
<tr>
<td>Distant (Bordes et al. 2011)</td>
<td>$|W_{rh} h - W_{rt} t|<em>1$, $W</em>{rh}, W_{rt} \in \mathbb{R}^{k \times k}$</td>
<td>$O(n_e k + 2n_r k^2)$</td>
</tr>
<tr>
<td>Bilinear (Jenatton et al. 2012)</td>
<td>$h^T W_r t$, $W_r \in \mathbb{R}^{k \times k}$</td>
<td>$O(n_e k + n_r k^2)$</td>
</tr>
<tr>
<td>Single Layer</td>
<td>$u_r^T f(W_{rh} h + W_{rt} t + b_r)$, $u_r, b_r \in \mathbb{R}^s$, $W_{rh}, W_{rt} \in \mathbb{R}^{s \times k}$</td>
<td>$O(n_e k + n_r (s k + s))$</td>
</tr>
<tr>
<td>NTN (Socher et al. 2013)</td>
<td>$u_r^T f(h^T W_r t + W_{rh} h + W_{rt} t + b_r)$, $u_r, b_r \in \mathbb{R}^s$, $W_r \in \mathbb{R}^{k \times k \times s}$, $W_{rh}, W_{rt} \in \mathbb{R}^{s \times k}$</td>
<td>$O(n_e k + n_r (s k^2 + 2sk + 2s))$</td>
</tr>
<tr>
<td>TransH</td>
<td>$|(h - w_r^T h w_r) + d_r - (t - w_r^T t w_r)|^2_2$, $w_r, d_r \in \mathbb{R}^k$</td>
<td>$O(n_e k + 2n_r k)$</td>
</tr>
</tbody>
</table>
Existing work

• Typically evaluate on Freebase and WordNet

<table>
<thead>
<tr>
<th>Data</th>
<th>WN18</th>
<th>FB15K</th>
<th>WN11</th>
<th>FB13</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Rel</td>
<td>18</td>
<td>1,345</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>#Ent</td>
<td>40,943</td>
<td>14,951</td>
<td>38,696</td>
<td>75,043</td>
</tr>
<tr>
<td>#Train</td>
<td>141,442</td>
<td>483,142</td>
<td>112,581</td>
<td>316,232</td>
</tr>
<tr>
<td>#Valid</td>
<td>5,000</td>
<td>50,000</td>
<td>2,609</td>
<td>5,908</td>
</tr>
<tr>
<td>#Test</td>
<td>5,000</td>
<td>59,071</td>
<td>10,544</td>
<td>23,733</td>
</tr>
</tbody>
</table>

Wang et al. (2008)
### Application 1: Triples completion

<table>
<thead>
<tr>
<th>Dataset</th>
<th>WN18</th>
<th>FB15k</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MEAN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Raw</td>
<td>Filt.</td>
</tr>
<tr>
<td><strong>Unstructured (Bordes et al. 2012)</strong></td>
<td>315</td>
<td>304</td>
</tr>
<tr>
<td><strong>RESCAL (Nickel, Tresp, and Kriegel 2011)</strong></td>
<td>1,180</td>
<td>1,163</td>
</tr>
<tr>
<td><strong>SE (Bordes et al. 2011)</strong></td>
<td>1,011</td>
<td>985</td>
</tr>
<tr>
<td><strong>SME (Linear) (Bordes et al. 2012)</strong></td>
<td>545</td>
<td>533</td>
</tr>
<tr>
<td><strong>SME (Bilinear) (Bordes et al. 2012)</strong></td>
<td>526</td>
<td>509</td>
</tr>
<tr>
<td><strong>LFM (Jenatton et al. 2012)</strong></td>
<td>469</td>
<td>456</td>
</tr>
<tr>
<td><strong>TransE (Bordes et al. 2013b)</strong></td>
<td>263</td>
<td><strong>251</strong></td>
</tr>
<tr>
<td>TransH (unif.)</td>
<td>318</td>
<td>303</td>
</tr>
<tr>
<td>TransH (bern.)</td>
<td>400.8</td>
<td>388</td>
</tr>
</tbody>
</table>

*Wang et al. (2008)*
Application 2: Triples classification

<table>
<thead>
<tr>
<th>Dataset</th>
<th>WN11</th>
<th>FB13</th>
<th>FB15k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distant Model</td>
<td>53.0</td>
<td>75.2</td>
<td>-</td>
</tr>
<tr>
<td>Hadamard Model</td>
<td>70.0</td>
<td>63.7</td>
<td>-</td>
</tr>
<tr>
<td>Single Layer Model</td>
<td>69.9</td>
<td>85.3</td>
<td>-</td>
</tr>
<tr>
<td>Bilinear Model</td>
<td>73.8</td>
<td>84.3</td>
<td>-</td>
</tr>
<tr>
<td>NTN</td>
<td>70.4</td>
<td>87.1</td>
<td>66.5 (≈ 40h)</td>
</tr>
<tr>
<td>TransE (unif.)</td>
<td>75.85</td>
<td>70.9</td>
<td>79.7 (≈ 5m)</td>
</tr>
<tr>
<td>TransE (bern.)</td>
<td>75.87</td>
<td>81.5</td>
<td>87.3 (≈ 5m)</td>
</tr>
<tr>
<td>TransH (unif.)</td>
<td>77.68</td>
<td>76.5</td>
<td>80.2 (≈ 30m)</td>
</tr>
<tr>
<td>TransH (bern.)</td>
<td>78.80</td>
<td>83.3</td>
<td>87.7 (≈ 30m)</td>
</tr>
</tbody>
</table>

Wang et al. (2008)
Code availability

• Code for replicating experiments can be found at https://github.com/glorotxa/SME; implemented using both theano/tensorflow backend

• Unclear how to extend to new, sparse data, how to scale to much bigger KGs
Application 3: ‘Featurizing’ locations

• E.g. Convering ‘locations’ into feature vectors
• Relevant for toponym resolution, building rich graphs...


https://github.com/mayankkejriwal/Geonames-embeddings
Features encode spatial proximity

• But could encode much else, lots of room for new research!
Embeddings and **extracted** knowledge graphs

- **Do embeddings work for extracted KGs?**
  - Approach by Pujara et al. (2017): Evaluate on the NELL knowledge graph, containing millions of candidates extracted from WWW text

- **Observations:**
  - Baseline (threshold input) wins against embeddings
  - Best results from graphical model (PSL-KGI) using rules & uncertainty
  - More complex embedding methods have the worst performance

- **Conclusion:** Embeddings have poor performance on sparse & noisy KGs extracted from text

- **Key question for future research:** How do we make embeddings work for extracted KGs?

### Table

<table>
<thead>
<tr>
<th>Method</th>
<th>AUC</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NELL</td>
<td>0.765</td>
<td>0.673</td>
</tr>
<tr>
<td>TransH</td>
<td>0.701</td>
<td>0.783</td>
</tr>
<tr>
<td>HoLE</td>
<td>0.710</td>
<td>0.783</td>
</tr>
<tr>
<td>TransE</td>
<td>0.726</td>
<td>0.783</td>
</tr>
<tr>
<td>STransE</td>
<td>0.784</td>
<td>0.783</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.873</td>
<td>0.828</td>
</tr>
<tr>
<td>PSL-KGI</td>
<td><strong>0.891</strong></td>
<td><strong>0.848</strong></td>
</tr>
</tbody>
</table>
Summary

• Knowledge graph embedding (KGE) is an active research area

• Uses machine learning and neural networks to ‘vectorize’ entities and relationships

• Implementations can be slow, recently this has started to change

• Unlike PSL, ecosystem not yet matured