Random Walk Inference and Learning on Knowledge Base

Ni Lao

DB Seminar
Mar 2, 2011
Outline

• The Need for Efficient Logical Inference
  – Open Domain Information Extraction
  – The NELL project
  – FOIL
  – Random Walk Inference

• Approach
  – Path Ranking Algorithm (Recap)
  – Efficient Random Walk (Recap)
  – Data-Driven Path Finding
  – Low-Variance Sampling

• Results
  – Cross Validation on the Training Queries
  – Evaluation by Mechanical Turk

• Conclusion
Open Domain Information Extraction

- **KnowItAll Project (Univ. W.)**
  - 0.5B facts extracted from 0.1B web pages

- **DBpedia (Univ. Leipzig)**
  - 3.5M entities 0.67B facts extracted from wikipedia

- **YAGO (Yet Another Great Ontology, Max-Planck-Institute)**
  - 2M entities 20M facts extracted from wikipedia and wordNet

- **NELL (Never-Ending Language Learning, CMU)**
  - close domain
  - 0.6M facts extracted from 1B webpages

- **IE projects in companies (e.g. Google)**
The Need for Inference

• Tasks
  – Discover new facts
    • finding new triples $R(x,y)$
  – Extend the ontology
    • discovery new relation $R$ or new entity type $T$

• Challenges
  – Uncertainty
    • extracted knowledge is incomplete and noise
  – Scalability
    • the size of knowledge base can be very large
The NELL Project

• Never-Ending Language Learning
  – develop a never-ending learning system that operates 24 hours per day, for years, to continuously improve its ability to read (extract structured facts from) the web (Carlson et al., 2010)

• close domain, semi-supervised
• combine multiple strategies:
  – word patterns, text context, html patterns, logic inference

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>cityInState</td>
<td>(troy, Michigan)</td>
</tr>
<tr>
<td>musicArtistGenre</td>
<td>(Nirvana, Grunge)</td>
</tr>
<tr>
<td>tvStationInCity</td>
<td>(WLS-TV, Chicago)</td>
</tr>
<tr>
<td>sportUsesEquip</td>
<td>(soccer, balls)</td>
</tr>
<tr>
<td>athleteInLeague</td>
<td>(Dan Fouts, NFL)</td>
</tr>
<tr>
<td>starredIn</td>
<td>(Will Smith, Seven Pounds)</td>
</tr>
<tr>
<td>productType</td>
<td>(Acrobat Reader, FILE)</td>
</tr>
<tr>
<td>athletePlaysSport</td>
<td>(scott shields, baseball)</td>
</tr>
<tr>
<td>cityInCountry</td>
<td>(Dublin Airport, Ireland)</td>
</tr>
</tbody>
</table>
Link Prediction Task

• We consider 48 relations which have more than 100 instances (links) in the knowledge base

• We create two tasks for each relation—i.e., predicting $y$ given $x$ and predicting $x$ given $y$
  – AthletePlaysInLeague(\text{HinesWard},?)
  – AthletePlaysInLeague(?,?,\text{NFL})

• Training data
  – the actual nodes $y$ known to satisfy $R(x; y)$ are treated as labeled positive examples, and any other nodes are treated as negative examples
First Order Inductive Learner

• FOIL (Quinlan and Cameron-Jones, 1993)
  – Given positive and negative examples of some concept and a set of background predicates, FOIL inductively generates a logical concept definition (or rule) for the concept.
  – The learning process is similar to that of decision trees, but in relational domain

• Examples
  + AthletePlaysInLeague(HinesWard,NFL)
  - AthletePlaysInLeague(HinesWard,NBA)

• Rules
  AthletePlaysForTeam(x; y)
  ^ TeamPlaysInLeague(y; z)
  ➔ AthletePlaysInLeague(x,z)
FOIL Problems

• Inference—horn clauses can be costly to evaluate
  – assumes that the predicates is functional
  – e.g. each athlete plays in at most one League

• Prediction—cannot combine multiple noisy rules (and facts)
  – e.g. rules for teamPlaysSports

\[
\begin{align*}
C \xrightarrow{teamAlsoKnownAs} C \\
C \xrightarrow{teamHomeStadium} C \\
C \xrightarrow{teamMember} C \\
C \xrightarrow{teamPlaysAgainstTeam} C \\
C \xrightarrow{teamPlaysSport} C \\
C \xrightarrow{stadiumHomeToSport} C \\
C \xrightarrow{athletePlaysSport} C
\end{align*}
\]
Random Walk Inference

• Consider a path from x to y via the sequence of edge types $isa, isa^{-1}$, $AthletePlaysInLeague$, which corresponds to the Horn clause $isa(x, c) \land isa(x', c) \land AthletePlaysInLeague(x', y) \rightarrow AthletePlaysInLeague(x; y)$

• We can combine such low precision rules with Path Constrained Random Walk models
Contributions

• Soft rules
  – combine low precision rules with learned logistic regression
  – double the precision at rank 100, and the new learning method is also applicable to many more inference tasks

• Efficiency
  – we can exploit efficient approximation schemes for random walks
  – RW based method trains in minutes compared to 1 week for FOIL based method
Outline

• The Need for Efficient Logical Inference
  – Open Domain Information Extraction
  – The NELL project
  – FOIL
  – Random Walk Inference

• Approach
  – Path Ranking Algorithm (Recap)
  – Efficient Random Walk (Recap)
  – Data-Driven Path Finding
  – Low-Variance Sampling

• Results
  – Cross Validation on the Training Queries
  – Evaluation by Mechanical Turk

• Conclusion
Path Constrained Random Walk

(Lao & Cohen, ECML 2010)

• A **Relation path** $P=(R_1, ..., R_n)$ is a sequence of relations
  
  – E.g. 
  
  $$
  C \xrightarrow{\text{athletePlaysForTeam}} C \xrightarrow{\text{teamPlaysSport}} C
  $$

• **Path Constrained Random Walk**

  – Recursively define a distribution for each path
  
  – $E_s$ is a set of seeds

\[
\begin{align*}
    h_{E_s,R_1...R_i+1}(e) &= \sum_{e' \in \text{range}(R_i)} h_{E_s,R_1...R_i}(e') P(e \mid e'; R_i) \\
    P(e \mid e'; R_i) &= \begin{cases}
      1 / \deg(e', R_i) & e' \xrightarrow{R_i} e \\
      0 & \text{otherwise}
    \end{cases}
\end{align*}
\]

\[
\begin{align*}
    h_{E_s,<\text{empty}>}(e) &= \begin{cases}
      1 / |E_s| & e \in E_s \\
      0 & \text{otherwise}
    \end{cases}
\end{align*}
\]
Path Ranking Algorithm (PRA)

(Lao & Cohen, ECML 2010)

• Given a set of seed nodes $E_q$, and a target type $T_q$, a PRA model ranks target entities by linearly combine the distributions of different paths

$$\text{score}(e; \theta) = \sum_{P \in \mathcal{P}(q, L)} h_P(e, E_q) \theta_P$$

- where $\mathcal{P}(q, L) = \{P\}$ is the set of all relation paths with range $T_q$ and length $\leq L$

• or in matrix form $s = A \theta$,

- where $A$ is the feature matrix. each column of $A$ is a sparse distribution produced by one of the relation paths we define each row of $A$ as $x_i$
Parameter Estimation

- Given a relation R and a set of node pairs \( \{(s_i, t_i)\} \)
  - we can construct a training dataset \( D = \{(x_i, y_i)\} \), where \( x_i \) is a vector of all the path features for \( (s_i, t_i) \), and \( y_i \) indicates whether \( R(s_i, t_i) \) is true

- Estimate \( \theta \) by maximizing a regularized log-likelihood
  \[
  O(\theta) = \sum_i o_i(\theta) - \lambda_1 |\theta|_1 - \lambda_2 |\theta|_2 / 2
  \]
  - per-query objective function
    \[
    o_i(\theta) = w_i [y_i \ln p_i + (1 - y_i) \ln(1 - p_i)]
    \]
  - predicted relevance
    \[
    p_i = p(y_i = 1| x_i ; \theta) = \frac{\exp(\theta^T x_i)}{1 + \exp(\theta^T x_i)}
    \]
Efficient Inference

- Exact random walk results in probability to many internal nodes on the graph
  - Computation should be focused on the nodes we care about
  - e.g. Query-Specific Inference (Chechetka & Guestrin, 2010) on graphical models

1 million nodes

query node

A few nodes that we care about

(Lao & Cohen, KDD 2010)
Efficient Inference

- Rational for sampling
  - a few random walkers are enough to distinguish good target nodes from bad target nodes

A few nodes that we care about

1 million nodes
Data-Driven Path Finding

- Impractical to enumerate all possible paths even for small path lengths

- To rules out the paths that cannot be grounded for many queries
  - define a query $s$ to be supporting a path $P$ if $h_{s,P}(e) \neq 0$ for any entity $e$
  - require a path to be supported by $\alpha$ portion of the training queries

- To rules out the paths which are not related to the task
  - require a path to retrieve at least one target entity in the training set

<table>
<thead>
<tr>
<th></th>
<th>$l=3$</th>
<th>$l=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>all paths up to length L</td>
<td>15,376</td>
<td>1,906,624</td>
</tr>
<tr>
<td>+query support $\geq \alpha = 0.1$</td>
<td>522</td>
<td>5016</td>
</tr>
<tr>
<td>+ever reach a target entity</td>
<td>136</td>
<td>792</td>
</tr>
<tr>
<td>+$L_1$ regularization</td>
<td>63</td>
<td>271</td>
</tr>
</tbody>
</table>

Table 2: Number of paths in PRA models of maximin path length 3 and 4. Averaged over 96 tasks.
Low-Variance Sampling

• Sampling introduce variance to the result distributions,
  – e.g. consider a node in the graph with just two out links with equal weights. Generate two walkers. With 50 percent chance both walkers will follow the same branch

• Low-Variance Sampling (LVS) (Thrun et al., 2005)
  – generate M correlated samples, by drawing a single number \( r \) from \((0,M^{-1})\)

\[
samples \text{ correspond to } \frac{kM^{-1}+r}{M}, \quad k=0..M-1
\]
• low-variance sampling can improve prediction for both finger printing and particle filtering
Outline

• The Need for Efficient Logical Inference
  – Open Domain Information Extraction
  – The NELL project
  – FOIL
  – Random Walk Inference

• Approach
  – Path Ranking Algorithm (Recap)
  – Efficient Random Walk (Recap)
  – Data-Driven Path Finding
  – Low-Variance Sampling

• Results
  – Cross Validation on the Training Queries
  – Evaluation by Mechanical Turk

• Conclusion
Cross Validation on the Training Queries

• compares three methods using 5-fold cross validation and the Mean Reciprocal Rank (MRR)
  – supervised training can significantly improve retrieval quality
  – path information can produce further improvement

Table 3: Compare PRA with RWR models. MRRs and training times are averaged over 96 tasks.

<table>
<thead>
<tr>
<th></th>
<th>( l=2 )</th>
<th>( l=3 )</th>
<th>( l=2 )</th>
<th>( l=3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR Training</td>
<td>MRR Training</td>
<td>MRR Training</td>
<td>MRR Training</td>
</tr>
<tr>
<td>RWR(no train)</td>
<td>0.271</td>
<td>0.456</td>
<td>0.280</td>
<td>0.471</td>
</tr>
<tr>
<td>RWR</td>
<td>0.280</td>
<td>3.7s</td>
<td>0.307</td>
<td>5.7s</td>
</tr>
<tr>
<td>PRA</td>
<td>0.307</td>
<td>5.7s</td>
<td>0.516</td>
<td>15.4s</td>
</tr>
</tbody>
</table>
Evaluation by Mechanical Turk

• There are many test queries per predicate
  – On average 7,000 test queries for each functional predicate, and 13,000 for each non-functional predicate

• We sort the queries for each predicate according to the scores of their top ranked results, and then evaluate the precisions at top 10, 100 and 1000 positions
  – randomly subsample 50/100, and 50/1000

• Each piece of belief is voted by 5 turkers
  – Workers are given assertions like “Hines Ward plays for the team Steelers”, as well as Google search links for each entity

<table>
<thead>
<tr>
<th>AMT=F</th>
<th>AMT=T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gold=F</td>
<td>25%</td>
</tr>
<tr>
<td>Gold=T</td>
<td>11%</td>
</tr>
</tbody>
</table>
Predicates with FOIL

• Using paired t-test at task level,
  – \( p\text{-value}=0.3 \) for \( p@10 \), \( p\text{-value}=0.003 \) for \( p@100 \)

<table>
<thead>
<tr>
<th>Task</th>
<th>( P_{majority} )</th>
<th>#Paths ( p@10 )</th>
<th>( p@100 )</th>
<th>( p@1000 )</th>
<th>#Rules</th>
<th>#Query ( p@10 )</th>
<th>( p@100 )</th>
<th>( p@1000 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>athletePlaysForTeam</td>
<td>0.07</td>
<td>125</td>
<td>0.4</td>
<td>0.46</td>
<td>0.66</td>
<td>1(+1)</td>
<td>7</td>
<td>0.6</td>
</tr>
<tr>
<td>athletePlaysInLeague</td>
<td>0.60</td>
<td>15</td>
<td>1.0</td>
<td>0.84</td>
<td>0.80</td>
<td>3(+30)</td>
<td>332</td>
<td>0.9</td>
</tr>
<tr>
<td>athletePlaysSport</td>
<td>0.73</td>
<td>34</td>
<td>1.0</td>
<td>0.78</td>
<td>0.70</td>
<td>2(+30)</td>
<td>224</td>
<td>1.0</td>
</tr>
<tr>
<td>stadiumLocatedInCity</td>
<td>0.05</td>
<td>18</td>
<td>0.9</td>
<td>0.62</td>
<td>0.54</td>
<td>1(+0)</td>
<td>25</td>
<td>0.7</td>
</tr>
<tr>
<td>teamHomeStadium</td>
<td>0.02</td>
<td>66</td>
<td>0.3</td>
<td>0.48</td>
<td>0.34</td>
<td>1(+0)</td>
<td>2</td>
<td>0.2</td>
</tr>
<tr>
<td>teamPlaysInCity</td>
<td>0.10</td>
<td>29</td>
<td>1.0</td>
<td>0.86</td>
<td>0.62</td>
<td>1(+0)</td>
<td>60</td>
<td>0.9</td>
</tr>
<tr>
<td>teamPlaysInLeague</td>
<td>0.26</td>
<td>36</td>
<td>1.0</td>
<td>0.70</td>
<td>0.64</td>
<td>4(+151)</td>
<td>30</td>
<td>0.9</td>
</tr>
<tr>
<td>teamPlaysSport</td>
<td>0.42</td>
<td>21</td>
<td>0.7</td>
<td>0.60</td>
<td>0.62</td>
<td>4(+86)</td>
<td>48</td>
<td>0.9</td>
</tr>
<tr>
<td><strong>average</strong></td>
<td>0.28</td>
<td>43</td>
<td>0.79</td>
<td>0.668</td>
<td>0.615</td>
<td></td>
<td>91</td>
<td>0.76</td>
</tr>
</tbody>
</table>
Non-functional Predicates

- Randomly sampled another 10 non functional predicates
  - slightly lower accuracy than functional predicates

<table>
<thead>
<tr>
<th>Task</th>
<th>$P_{majority}$</th>
<th>#Paths</th>
<th>p@10</th>
<th>p@100</th>
<th>p@1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>teamMember</td>
<td>0.01</td>
<td>203</td>
<td>0.8</td>
<td>0.64</td>
<td>0.48</td>
</tr>
<tr>
<td>companiesHeadquarteredHere</td>
<td>0.05</td>
<td>42</td>
<td>0.6</td>
<td>0.54</td>
<td>0.60</td>
</tr>
<tr>
<td>publicationJournalist</td>
<td>0.02</td>
<td>25</td>
<td>0.7</td>
<td>0.70</td>
<td>0.64</td>
</tr>
<tr>
<td>producedBy</td>
<td>0.19</td>
<td>13</td>
<td>0.5</td>
<td>0.58</td>
<td>0.68</td>
</tr>
<tr>
<td>competesWith</td>
<td>0.19</td>
<td>74</td>
<td>0.6</td>
<td>0.56</td>
<td>0.72</td>
</tr>
<tr>
<td>hasOfficeInCity</td>
<td>0.03</td>
<td>262</td>
<td>0.9</td>
<td>0.84</td>
<td>0.60</td>
</tr>
<tr>
<td>journalistWritesForPublication</td>
<td>0.13</td>
<td>9</td>
<td>0.7</td>
<td>0.74</td>
<td>0.78</td>
</tr>
<tr>
<td>professionIsTypeOfProfession</td>
<td>0.32</td>
<td>4</td>
<td>0.5</td>
<td>0.66</td>
<td>0.80</td>
</tr>
<tr>
<td>teamWonTrophy</td>
<td>0.24</td>
<td>56</td>
<td>0.5</td>
<td>0.50</td>
<td>0.46</td>
</tr>
<tr>
<td>worksFor</td>
<td>0.13</td>
<td>62</td>
<td>0.6</td>
<td>0.60</td>
<td>0.74</td>
</tr>
<tr>
<td><strong>average</strong></td>
<td>0.13</td>
<td>75</td>
<td>0.64</td>
<td>0.636</td>
<td>0.65</td>
</tr>
</tbody>
</table>
### Example Rules

Table 6: The rank of PRA paths which correspond to N-FOIL rules sorted by decreasing path weights.

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Rank</th>
<th>( \hat{P} )</th>
<th>( N_+ )</th>
<th>( N_- )</th>
<th>N-FOIL Rule in Path Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>athletePlaysForTeam</td>
<td>none</td>
<td>0.63</td>
<td>13</td>
<td>6</td>
<td>( c \xrightarrow{\text{athleteCoach}} c \xrightarrow{\text{coachesTeam}} c )</td>
</tr>
<tr>
<td>athletePlaysInLeague</td>
<td>none</td>
<td>0.86</td>
<td>11</td>
<td>0</td>
<td>( c \xrightarrow{\text{athletePlaysForTeam}} c \xrightarrow{\text{teamPlaysInLeague}} c )</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.95</td>
<td>1134</td>
<td>60</td>
<td>( c \xrightarrow{\text{teammate}} c \xrightarrow{\text{athletePlaysInLeague}} c )</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>0.90</td>
<td>26</td>
<td>1</td>
<td>( c \xrightarrow{\text{teammate}} c \xrightarrow{\text{athletePlaysInLeague}} c )</td>
</tr>
<tr>
<td>athletePlaysSport</td>
<td>5</td>
<td>0.89</td>
<td>1102</td>
<td>128</td>
<td>( c \xrightarrow{\text{athletePlaysForTeam}} c \xrightarrow{\text{teamPlaysSport}} c )</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>0.91</td>
<td>27</td>
<td>1</td>
<td>( c \xrightarrow{\text{teammate}} c \xrightarrow{\text{athletePlaysSport}} c )</td>
</tr>
<tr>
<td>stadiumLocatedInCity</td>
<td>3</td>
<td>0.88</td>
<td>169</td>
<td>22</td>
<td>( c \xrightarrow{\text{stadiumHomeTeam}} c \xrightarrow{\text{teamPlaysInCity}} c )</td>
</tr>
<tr>
<td>teamHomeStadium</td>
<td>none</td>
<td>0.60</td>
<td>27</td>
<td>16</td>
<td>( c \xrightarrow{\text{teamAlsoKnownAs}} c \xrightarrow{\text{teamHomeStadium}} c )</td>
</tr>
<tr>
<td>teamPlaysInCity</td>
<td>1</td>
<td>0.88</td>
<td>250</td>
<td>33</td>
<td>( c \xrightarrow{\text{teamHomeStadium}} c \xrightarrow{\text{stadiumLocatedInCity}} c )</td>
</tr>
<tr>
<td>teamPlaysInLeague</td>
<td>none</td>
<td>0.93</td>
<td>49</td>
<td>2</td>
<td>( c \xrightarrow{\text{teamAlsoKnownAs}} c \xrightarrow{\text{teamPlaysInLeague}} c )</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>0.76</td>
<td>35</td>
<td>9</td>
<td>( c \xrightarrow{\text{teamHomeStadium}} c \xrightarrow{\text{stadiumHomeToLeague}} c )</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.94</td>
<td>478</td>
<td>27</td>
<td>( c \xrightarrow{\text{teamMember}} c \xrightarrow{\text{athletePlaysInLeague}} c )</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.82</td>
<td>2751</td>
<td>621</td>
<td>( c \xrightarrow{\text{teamPlaysAgainstTeam}} c \xrightarrow{\text{teamPlaysInLeague}} c )</td>
</tr>
<tr>
<td>teamPlaysSport</td>
<td>none</td>
<td>0.86</td>
<td>43</td>
<td>5</td>
<td>( c \xrightarrow{\text{teamAlsoKnownAs}} c \xrightarrow{\text{teamPlaysSport}} c )</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>0.69</td>
<td>63</td>
<td>26</td>
<td>( c \xrightarrow{\text{teamHomeStadium}} c \xrightarrow{\text{stadiumHomeToSport}} c )</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.91</td>
<td>487</td>
<td>47</td>
<td>( c \xrightarrow{\text{teamMember}} c \xrightarrow{\text{athletePlaysSport}} c )</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.73</td>
<td>2635</td>
<td>962</td>
<td>( c \xrightarrow{\text{teamPlaysAgainstTeam}} c \xrightarrow{\text{teamPlaysSport}} c )</td>
</tr>
</tbody>
</table>
Conclusion

• Contributions
  – Combine low precision rules with learned logistic regression
  – We can exploit efficient approximation schemes for random walks

• Future work
  – Inference starting from both the query nodes and target can be efficient in discovering long paths
  – Inference starting from the target nodes of training queries is a potential way to discover specialized paths (with constant nodes)
  – Generalizing inference paths to inference trees or graphs can produce more expressive random walk inference models