Fall Semester Summary

Chenguang Wang
Previous studies: *Schema-simple HINs*
- DBLP network: four entity types (Paper, Author, Venue, Term), and several relation types

In real world: *Schema-rich HINs*
- Networks have more complex schema
  - Freebase network: 1,500+ entity types and 35,000+ relation types

- In *schema-simple* HINs
  - Basic functions like similarity search works well
  - Reason: user provides meta-paths to express interest
  - *Author-Paper-Venue-Paper-Author*

  - find similar authors publishing papers at the same venue

- In *schema-rich* HINs
  - Unrealistic: too many possible meaningful meta-paths
  - Reason: especially for the long and sophisticated ones
Relation Similarity Search

(Joint work with Yizhou Sun, Yanglei Song, Jiawei Han, Yangqiu Song)

Latent Semantic Relations

- $P_1$: co-founders (0.45)
- $P_2$: schoolmates (0.25)
- $P_3$: co-inventors (0.15)

Query

Larry Page  Sergey Brin

Jerry Yang  David Filo

Search Result (ranked)

Bill Gates  Paul Allen

Steve Jobs  Steve Wozniak

Steve Ballmer  Mark Zuckerberg

......
Relation Similarity Search Framework

Query e.g. \( Q = \{<\text{Larry Page, Sergey Brin}>, <\text{Jerry Yang, David Filo}>\} \)

1. Query-based Network Schema Generation

2. Query-based Meta-Paths Generation

Query-based Meta-Paths

4. Optimization Model

4. Optimization Model

3. Criteria-based Selection

Filtered Meta-Paths

Weighted Meta-Paths

5. RelSim-based Search Algorithm

Ranked Relation Instances
<table>
<thead>
<tr>
<th>Query: {\langle Google, Larry Page\rangle, \langle Microsoft, Bill Gates\rangle, etc.}</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Organizations</strong> is founded by <strong>Founder</strong></td>
</tr>
<tr>
<td><strong>Organization</strong> run business in <strong>Industry</strong> win award in^{-1} <strong>Founder</strong></td>
</tr>
<tr>
<td><strong>Organization</strong> is founded by <strong>Person</strong> is influence peer^{-1} <strong>Founder</strong></td>
</tr>
<tr>
<td><strong>Organization</strong>'s leadership <strong>Person</strong> mailing address^{-1} <strong>Location</strong> mailing address^{-1} <strong>Founder</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query: {\langle Google, Larry Page\rangle, \langle Yahoo!, Marissa Mayer\rangle, etc.}</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Organization</strong> run by <strong>CEO</strong> job title <strong>Founder</strong></td>
</tr>
<tr>
<td><strong>Organization</strong> founded date <strong>Date</strong> graduation date^{-1} <strong>Founder</strong></td>
</tr>
<tr>
<td><strong>Organization</strong> headquarter <strong>Location</strong> education institute <strong>Founder</strong></td>
</tr>
<tr>
<td><strong>Organization</strong> run business in <strong>Industry</strong> win award in^{-1} <strong>Founder</strong></td>
</tr>
</tbody>
</table>
“Schema-Rich” Raises Challenge (2): Semantically Similar Relations

- In schema-rich HINs, differently expressed relations are in fact semantically similar
  - 
  - \((X, \text{wrote}, Y)\) and \((X, \text{‘s written work}, Y)\)
  - \((X, \text{is founder of}, Y)\) and \((X, \text{is CEO of}, Y)\)
  - \((X, \text{written by}, Y)\) and \((X, \text{part of}, Z)^{Y, \text{wrote}, Z}\)

- Grouping semantically similar relations into canonical clusters would facilitate and improve many applications
  - knowledge base completion, information extraction, information retrieval, and more
Relation Canonicalization

Unstructured Data

"In 1995, J.K. Rowling finished writing manuscript for Harry Potter and the Philosopher's Stone on an old manual typewriter."

"Google was founded by Larry Page and Sergey Brin while they were Ph.D. students at Stanford University."

Knowledge Bases

Cluster 1
- J.K. Rowling  Philosopher's Stone
- J.K. Rowling  Harry Potter Series
- Philosopher's Stone  Harry Potter Series

Cluster 2
- Google  Larry Page
- Larry Page  Google

Canonicalizing Relations (CTGC)

Multi-Hop Relation Generation

Open Information Extraction
Canonicalizing Open Domain Relations

(Joint work with Yangqiu Song, Chi Wang, Jiawei Han, Heng Ji, Dan Roth)
Submitted to WWW’15
Experiments

The graph shows the performance of different clustering algorithms as a function of the number of relation constraints. The x-axis represents the number of relation constraints, while the y-axis represents the Normalized Mutual Information (NMI). The lines and markers correspond to different algorithms:

- Kmeans
- CKmeans
- ITCC
- CITCC
- TFBC
- TGC
- CTGC

As the number of relation constraints increases, the NMI for all algorithms generally increases as well, indicating improved clustering performance.
Experiments
<table>
<thead>
<tr>
<th>Relationship</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization-Founder</td>
<td>(X, founded(^{-1}), Y); (X, was founded by(^{-1}), Y); (X, directed by, Y); (X, , led by, Y); (X, is established by, Y);</td>
</tr>
<tr>
<td>Book-Author</td>
<td>(X, wrote(^{-1}), Y); (X, is a play by, Y); (X, is a book by, Y); (X, renamed to(^{-1}), Y); (X, is a poem by, Y);</td>
</tr>
<tr>
<td>Actor-Film</td>
<td>(X, star(^{-1}), Y); (X, feature(^{-1}), Y); (X, stars(^{-1}), Y); (X, who played, Y); (X, starred in, Y); (X, ’s capital in, Y);</td>
</tr>
<tr>
<td>Location-Contains</td>
<td>(X, locate capital in, Y); (X, build, Y); (X, is contained by(^{-1}), Y); (X, have, Y); (X, extend, Y); (X, contains, Y);</td>
</tr>
<tr>
<td>Music-Track</td>
<td>(X, released, Y); (X, containing, Y); (X, has a song, Y); (X, from(^{-1}), Y); (X, is popular in(^{-1}), Y); (X, is in a concert, Y);</td>
</tr>
<tr>
<td>Person-Profession</td>
<td>(X, is good at, Y); (X, referred to, Y); (X, major in, Y); (X, is a celebrity, Y); (X, is talent in, Y); (X, is a profession in, Y);</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization-Founder</td>
<td>(X, founded by, Y); (X, led by, Y); (X, is the owner of(^{-1}), Y); (X, , sold by, Y); (X, , owned by, Y); (X, is in, Y);</td>
</tr>
<tr>
<td>Book-Author</td>
<td>(X, is the author of(^{-1}), Y); (X, written by, Y); (X, edited by, Y); (X, composed by, Y); (X, is a poem composed(^{-1}), Y);</td>
</tr>
<tr>
<td>Actor-Film</td>
<td>(X, , which stars(^{-1}), Y); (X, act in, Y); (X, makes a brief appearance, Y); (X, , appears in, Y); (X, performs, Y);</td>
</tr>
<tr>
<td>Location-Contains</td>
<td>(X, locate capital in, Y); (X, ’s capital in, Y) (X, is a department of(^{-1}), Y); (X, is a state of(^{-1}), Y);</td>
</tr>
<tr>
<td>Music-Track</td>
<td>(X, released, Y); (X, containing, Y); (X, was released in(^{-1}), Y); (X, is recorded in(^{-1}), Y); (X, , a record, Y);</td>
</tr>
<tr>
<td>Person-Profession</td>
<td>(X, legend(^{-1}), Y); (X, retires from, Y); (X, ’s profession is, Y); (X, is famous in, Y); (X, win championship, Y);</td>
</tr>
</tbody>
</table>
Future Directions

- Generalization: Documents+TopMine->Open domain (schema-rich) network relation similarity search.

- Meta-Path Generation for schema-rich heterogeneous information networks.

- RelSim-based Recommendation in schema-rich heterogeneous information networks.

- RelSim-based Clustering/Classification in schema-rich heterogeneous information networks.
Thanks for all your great help!