Wrappers for Web Data Extraction
Extracting Data from Semi-structured Sources

NAME: Casablanca Restaurant
STREET: 220 Lincoln Boulevard
CITY: Venice
PHONE: (310) 392-5751
Approaches to Wrapper Construction

• Manual Wrapper Construction
• Learning Wrappers from Labelled Examples
• Grammar Induction for Automatic Wrapper Construction
Grammar Induction Approach

- Pages automatically generated by scripts that encode results of db query into HTML
  - Script = grammar
- Given a set of pages generated by the same script
  - Learn the grammar of the pages
    - Wrapper induction step
  - Use the grammar to parse the pages
    - Data extraction step
RoadRunner

[Crescenzi, Mecca, & Merialdo]

- Automatically generates a wrapper from large web pages
  - Pages of the same class
  - No dynamic content from javascript, ajax, etc
- Infers source schema
  - Supports nested structures and lists
  - Extracts data from pages
- Efficient approach to large, complex pages with regular structure
Example Pages

- Compares two pages at a time to find similarities and differences
- Infers nested structure (schema) of page
- Extracts fields
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th></th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>This book introduces the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>reader to the theory and technology...[TRUNCATED]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>First Edition, Paperback</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1988</td>
<td></td>
<td></td>
<td>$20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Second Edition, Hard Cover</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2000</td>
<td></td>
<td></td>
<td>$30</td>
</tr>
<tr>
<td>Computer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>An undergraduate level</td>
</tr>
<tr>
<td>Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>introduction to computer...[TRUNCATED]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>First Edition, Paperback</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1986</td>
<td></td>
<td></td>
<td>$40</td>
</tr>
<tr>
<td>Paul Jones</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A comprehensive description</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>of XML and all related standards...[TRUNCATED]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XML at Work</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>First Edition, Paperback</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1999</td>
<td></td>
<td></td>
<td>$30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HTML and Scripts</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>half</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1993</td>
<td></td>
<td></td>
<td>$30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Second Edition, Hard Cover</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1999</td>
<td></td>
<td></td>
<td>$45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>JavaScripts</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>half</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2000</td>
<td></td>
<td></td>
<td>$50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A, must in every</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Webmaster's bookshelf...</td>
</tr>
</tbody>
</table>
Union-Free Regular Expression (UFRE)

- Web page structure can be represented as Union-Free Regular Expression (UFRE)
- UFRE is Regular Expressions without disjunctions
- If $a$ and $b$ are UFRE, then the following are also UFREs
  - $a.b$
  - $(a)^+$
  - $(a)?$
Union-Free Regular Expression (UFRE)

- Web page structure can be represented as *Union-Free Regular Expression (UFRE)*
- UFRE is Regular Expressions without *disjunctions*
- If *a* and *b* are UFRE, then the following are also UFREs
  - *a.b* → string fields
  - *(a)*+ → lists (possibly nested)
  - *(a)*? → optional fields
- Strong assumption that usually holds
Approach

• Given a set of example pages
• Generate the *Union-Free Regular Expression* which contains example pages
• Find the least upper bounds on the RE lattice to generate a wrapper in *linear time*
• Reduces to finding the least upper bound on two UFREs
Matching/Mismatches

Given a set of pages of the same type

- Take the first page to be the *wrapper* (UFRE)
- Match each successive sample page against the wrapper
- *Mismatches* result in generalizations of wrapper
  - String mismatches
  - Tag mismatches
Matching/Mismatches

Given a set of pages of the same type
• Take the first page to be the wrapper (UFRE)
• Match each successive sample page against the wrapper
• Mismatches result in generalizations of wrapper
  • String mismatches
    • Discover fields
  • Tag mismatches
    • Discover optional fields
    • Discover iterators
Example Matching

- Wrapper (initially Page 1):
  01: <HTML>
  02: Books of:
  03: <B>
  04: John Smith
  05: </B>
  06: <UL>
  07: <LI>
  08-10: <I>Title:</I>
  11: DB Primer
  12: </LI>
  13: <LI>
  14-16: <I>Title:</I>
  17: Comp. Sys.
  18: </LI>
  19: </UL>
  20: </HTML>

- Sample (Page 2):
  01: <HTML>
  02: Books of:
  03: <B>
  04: Paul Jones
  05: </B>
  06: <IMG src=.../>
  07: <UL>
  08: <LI>
  09-11: <I>Title:</I>
  12: XML at Work
  13: </LI>
  14: <LI>
  15-17: <I>Title:</I>
  18: HTML Scripts
  19: </LI>
  20: </UL>

- Wrapper after solving mismatches:

```
<HTML>Books of:<B>#PCDATA</B>
( <IMG src=.../> )?
<UL>
 ( <LI><I>Title:</I>#PCDATA</LI> )+
</UL></HTML>
```
String Mismatches: Discovering Fields

• String mismatches are used to discover fields of the document
• Wrapper is generalized by replacing “John Smith” with #PCDATA

<HTML>Books of: <B>John Smith</B> → <HTML> Books of: <B>#PCDATA</B>
Example Matching

- Wrapper (initially Page 1):

```
01: <HTML>
02: Books of:
03: <B>
04: John Smith
05: </B>
06: <UL>
07: <LI>
08-10: <I>Title:</I>
11: DB Primer
12: </LI>
13: <LI>
14-16: <I>Title:</I>
17: Comp. Sys.
18: </LI>
19: </UL>
20: </HTML>
```

Parsing

String mismatch (#PCDATA)

Tag mismatch (?)

Tag mismatch (+)

Terminal tag search and square matching

- Sample (Page 2):

```
01: <HTML>
02: Books of:
03: <B>
04: Paul Jones
05: </B>
06: <IMG src=.../>
07: <UL>
08: <LI>
09-11: <I>Title:</I>
12: XML at Work
13: </LI>
14: <LI>
15-17: <I>Title:</I>
18: HTML Scripts
19: </LI>
20: <LI>
21-23: <I>Title:</I>
24: Javascript
25: </LI>
26: </UL>
27: </HTML>
```
Tag Mismatches: Discovering Optionals

- First check to see if mismatch is caused by an iterator (described next)
- If not, could be an optional field in wrapper or sample
- Cross search used to determine possible optionals
- Image field determined to be optional:
  - ( <img src=.../> )?
Example Matching

- Wrapper (initially Page 1):

01:  <HTML>
02:  Books of:
03:  <B>
04:  John Smith
05:  </B>
06:  <UL>
07:  <LI>
08-10:  <I>Title:</I>
11:  DB Primer
12:  </LI>
13:  <LI>
14-16:  <I>Title:</I>
17:  Comp. Sys.
18:  </LI>
19:  </UL>
20:  </HTML>

- Sample (Page 2):

01:  <HTML>
02:  Books of:
03:  <B>
04:  Paul Jones
05:  </B>
06:  <IMG src=.../>
07:  <UL>
08:  <LI>
09-11:  <I>Title:</I>
12:  XML at Work
13:  </LI>
14:  <LI>
15-17:  <I>Title:</I>
18:  HTML Scripts
19:  </LI>
20:  <LI>
21-23:  <I>Title:</I>
24:  Javascript
25:  </LI>
26:  </UL>
27:  </HTML>

String Mismatch

String Mismatch

- Wrapper after solving mismatches:

<HTML>Books of:<B>PCDATA</B>
( <IMG src=.../>)?
<UL>
( <LI><I>Title:</I>PCDATA</LI>)+
</UL></HTML>
Tag Mismatches: Discovering Iterators

- Assume mismatch is caused by repeated elements in a list
  - End of the list corresponds to last matching token: </LI>
  - Beginning of list corresponds to one of the mismatched tokens: <LI> or </UL>
  - These create possible “squares”
- Match possible squares against earlier squares
- Generalize the wrapper by finding all contiguous repeated occurrences:
  - ( <LI><I>Title:</I>#PCDATA</LI> )+
Example Matching

- Wrapper (initially Page 1):

01:  <HTML>
02:  Books of:
03:  <B>
04:  John Smith
05:  </B>
06:  <UL>
07:  <LI>
08-10:  <I>Title:</I>
11:  DB Primer
12:  </LI>
13:  <LI>
14-16:  <I>Title:</I>
17:  Comp. Sys.
18:  </LI>
19:  </UL>
20:  </HTML>

- Sample (Page 2):

01:  <HTML>
02:  Books of:
03:  <B>
04:  Paul Jones
05:  </B>
06:  <IMG src=.../>
07:  <UL>
08:  <LI>
09-11:  <I>Title:</I>
12:  XML at Work
13:  </LI>
14:  <LI>
15-17:  <I>Title:</I>
18:  HTML Scripts
19:  </LI>
20:  <LI>
21-23:  <I>Title:</I>
24:  Javascript
25:  </LI>
26:  </UL>
27:  </HTML>

- Wrapper after solving mismatches:

<HTML>Books of:<B>#PCDATA</B>
 ( <IMG src=.../> )?
<UL>
 ( <LI><I>Title:</I>#PCDATA</LI> )+
</UL></HTML>
Internal Mismatches

• Generate *internal mismatch* while trying to match square against earlier squares on the same page

• Solving internal mismatches yield further refinements in the wrapper
  • List of book editions
  • *Special!*
Recursive Example

- Wrapper (initially Page 1):
  01-05: <HTML>Books of: <B>John Smith</B>
  06:  <UL>
  07:  <LI>
  08:  Computer Systems
  09:  <P>
  10:  <B>
  11:  1st Ed., 1995
  12:  </B>
  13:  </P>
  14:  </LI>
  15:  <LI>
  16:  Database Primer
  17:  <P>
  18:  <B>
  20-22:  <I>Special!</I>
  23:  </B>
  24:  </B>
  25:  2nd Ed., 2000
  26:  </B>
  27:  </P>
  28:  </LI>
  29-30:  </UL></HTML>

- Wrapper after solving mismatches:
  <HTML>Books of: <B>&#PCDATA</B>
  <UL>(<B>&#PCDATA</B>
   (</B>&#PCDATA</I>Special!</I>)?</I>
   </B>)+ </P>(</LI>)+
  </UL></HTML>
Discussion

- Assumptions:
  - Pages are well-structured
  - Structure can be modeled by UFRE (no disjunctions)

- Search space for explaining mismatches is huge
  - Uses a number of heuristics to prune space
    - Limited backtracking
    - Limit on number of choices to explore
    - Patterns cannot be delimited by optionals
  - Will result in pruning possible wrappers
Limitations

• Learnable grammars
  • Union-Free Regular Expressions (RoadRunner)
    • Variety of schema structure: tuples (with optional attributes) and lists of (nested) tuples
    • Does not efficiently handle disjunctions – pages with alternate presentations of the same attribute
  • Context-free Grammars
    • Limited learning ability
• User needs to provide a set of pages of the same type
Inferlink Web Extraction Software
FOR SALE: STOEGER M3500

post id: 4700468

Price: $500
Seller: Private Party
Account: Registered on 5/9/2013

Listings by this user

Listed On: Thursday, September 17, 2015
Listed In: Shotguns
Location: Keenesburg, Denver, Colorado - Map
Shipping: No

Manufacturer: Stoeger
Caliber: 12 Gauge
Action: Semi-automatic
Firearm Type: Shotgun

I have a Stoeger m3500. It is a year old. It has 200 rounds through it from clay shooting. Its in perfect condition. If you have any questions email or text me. 9703427061. I’m asking 500
FOR SALE: STOEGER M3500

post id: 4700468

Price: $500
Seller: Private Party
Account: Registered on 5/9/2013

Listed On: Thursday, September 17, 2015
Listed In: Shotguns
Location: Keenesburg, Denver, Colorado
Shipping: No

Manufacturer: Stoeger
Caliber: 12 Gauge
Action: Semi-automatic
Firearm Type: Shotgun

I have a Stoeger m3500. It is a year old. It has 200 rounds through it from clay shooting. It's in perfect condition. If you have any questions email or text me. 9703427061. I'm asking 500
Automated Extraction
[Minton et al., Inferlink]

- Title
- Description
- Seller
- Post Date
- Expiry Date
- Price
- Location
- Category
- Member Since
- Num Views
- Post ID
Automated Extraction

Input: A Pile of Pages
Automated Extraction

input: a pile of pages

Classify by Templates

pages clustered by template
Clustering

• Cluster
  • Based on the visible text
  • Page is broken into chunks
    • These are continuous blocks of text
  • Search for common visible chunks
    • Remove chunks that occur in all pages
    • Remove those that occur in fewer than 10 pages
  • Greedy algorithm to cluster the pages based on the remaining chunks
    • Sort by the size of the clusters created by each chunk
Automated Extraction

input: a pile of pages

Classify by Templates

pages clustered by template

Infer Extractor
Infer Extractor
Infer Extractor
Infer Extractor

extractor
Extractor Learning

- Input: cluster

- Select 5 random pages to build a template
  - Tokenize on space & punctuation
  - Start with n-grams of tuples of size n
    - Find those n-grams that occur on all pages
    - Keep only those n-grams that occur exactly once per page
    - Decompose pages based on these n-grams
    - Run algorithm recursively on decomposed page
  - Repeat above for size n-1 down to n=2
  - Construct template based on the decomposition
## Unsupervised Extraction Tool

### InferLink {Landmark}

<table>
<thead>
<tr>
<th>Code1392</th>
<th>0036</th>
<th>37122</th>
<th>37201</th>
<th>37122</th>
<th>37211</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Winchester model 140 12 gauge semiauto. 28inch ribbed and vented barrel with modified choke. Gun fires and cycles well. Text for pics.</td>
<td>Beretta hand gun for sale at a very good price with a delivery to any interested buyer in the state. contact me at <a href="mailto:silven2016@yandex.com">silven2016@yandex.com</a> or text me at</td>
<td>1956 Mossin nagant. 7.62&amp;215;54 bolt action Russian war rifle. Missing bayonet but rifle fires and cycles well.</td>
<td>Blue, 9½” barrel, original walnut grips, NcStar 4x32E scope/mount + lighted reticle and lens covers, 2 1/2 lb trigger (professional job), PRO-TECH OUTDOORS cordura shoulder/belt holster, original plastic box/lock &amp; key/.22LR cylinder/factory test envelope w/fired casing/instruction manual/original rear sight/ original trigger spring/protection gun sleeve. Everything appears as new except slight .22WMR cylinder spin mark with only 1 to 1½ 50-round boxes fired. .22LR cylinder never used. Scope is set for 50 yds. using Hornady 45 gr FTX .22WMR ammo. Total package f.o.b. Nashville, TN, $539. Must have TN driver's license and/ or TN CCW permit, or delivery to FFL required with photo of Driver's License and cleared payment.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expires1366</th>
<th>45 days, 23 hours</th>
<th>This ad has expired</th>
<th>46 days, 4 hours</th>
<th>89 days, 7 hours</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Facebook2081</th>
<th>2015</th>
<th>2015</th>
<th>2015</th>
<th>2016</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Firearms1276</th>
<th>raquo; Shotguns » Semi-Auto » Beretta hand gun for sale</th>
<th>raquo; PISTOLS » Bolt Action » 1956 Mossin nagant</th>
<th>raquo; PISTOLS » Revolver » Ruger New Model Single-Six 9 1/2&amp;215;8243;</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Group2122</th>
<th>Non-Felon / Legal</th>
<th>Non-Felon / Legal</th>
<th>Non-Felon / Legal</th>
<th>Mentally Capable / Non-Felon / Legal</th>
</tr>
</thead>
</table>

| ID1421 | 47955db13ce85f8e | 930552dc84196efa | 8655da9d2455236 | 6256c6aa4070496 |
# Extraction Evaluation

10 websites, 5 pages each

<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Desc</th>
<th>Seller</th>
<th>Date</th>
<th>Price</th>
<th>Loc</th>
<th>Cat</th>
<th>Member Since</th>
<th>Expires</th>
<th>Views</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Perfect</strong></td>
<td>1.0 (50/50)</td>
<td>.76 (37/49)</td>
<td>.95 (40/42)</td>
<td>.83 (40/48)</td>
<td>.87 (39/45)</td>
<td>.51 (23/45)</td>
<td>.68 (34/50)</td>
<td>1.0 (35/35)</td>
<td>.52 (15/29)</td>
<td>.76 (19/25)</td>
<td>.97 (35/36)</td>
</tr>
<tr>
<td><strong>Including partial and extra data</strong></td>
<td>1.0 (50/50)</td>
<td>.98 (48/49)</td>
<td>.95 (40/42)</td>
<td>.83 (40/48)</td>
<td>.98 (44/45)</td>
<td>.84 (38/45)</td>
<td>.88 (44/50)</td>
<td>1.0 (35/35)</td>
<td>.55 (16/29)</td>
<td>1.0 (25/25)</td>
<td>1.0 (36/36)</td>
</tr>
</tbody>
</table>
Discussion

- Inferlink approach solves some of the key limitations of Roadrunner
  - Pages do not all have to be of the same type
  - Multiple optionals would be treated as different page types
  - Scales well with complex pages
Web Data Extraction Software

- Beautiful Soup
  - [http://www.crummy.com/software/BeautifulSoup/](http://www.crummy.com/software/BeautifulSoup/)
  - Python library to manually write wrappers
- Jsoup
  - [http://jsoup.org/](http://jsoup.org/)
  - Java library to manually write wrappers
- ScrapingHub
  - [http://scrapinghub.com/](http://scrapinghub.com/)
  - Portia provides a wrapper learner
- Others
  - Tell us if you find a good one!
Aligning and Integrating Data in Karma
Karma

Interactive tool for rapidly extracting, cleaning, transforming, integrating and publishing data

http://www.isi.edu/integration/karma

@KarmaSemWeb
Information Integration in Karma

Karma semi-automatically generates Source Mappings

Domain Model

Samples of Source Data
Information Integration in Karma

Domain Model

Karma

Karma semi-automatically generates Source Mappings

Samples of Source Data

Karma supports multiple integration regimes
Secret Sauce: Karma Understands Your Data

Karma semi-automatically builds a semantic model of your data

Domain Model

Semantic Model of the Data

Source Mappings

Samples of Source Data
What is a Semantic Model?

Describe sources using classes & relationships in an ontology

<table>
<thead>
<tr>
<th>Source</th>
<th>name</th>
<th>date</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>2</td>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
</tbody>
</table>

Domain Model

- Person
  - name
  - birthdate
  - worksFor
  - ceo
  - livesIn

- Organization
  - name
  - phone

- Event
  - title
  - startDate
  - endDate

- Place
  - name
  - postalCode
  - nearby
  - isPartOf

- City
  - state
  - location

- State
  - state

Object property
- SubClassOf
Data property
- object property
- data property
### Semantic Types

<table>
<thead>
<tr>
<th>name</th>
<th>date</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
</tbody>
</table>

- **Person**: `name`, `birthdate`
- **City**: `name`
- **State**: `name`
- **Organization**: `name`
Relationships

<table>
<thead>
<tr>
<th>name</th>
<th>date</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
</tbody>
</table>
Semantic models will be formalized as Source Mappings

Key ingredient to automate source discovery, data integration, and publishing semantic data (RDF triples)
Karma uses **semantic models** to create knowledge graphs.
Karma semi-automatically builds semantic models

Karma uses semantic models to create knowledge graphs
Karma semi-automatically builds semantic models... and provides a nice GUI to edit them.

Karma uses semantic models to create knowledge graphs.
Semi-automatically Building Semantic Models in Karma
Approach
[Knoblock et al, ESWC 2012]

Sample Data

Domain Ontology

Learn Semantic Types

Construct a Graph

Steiner Tree

Extract Relationships
Example

Source

<table>
<thead>
<tr>
<th>name</th>
<th>date</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
</tbody>
</table>

Domain Ontology

Find a semantic model for the source (map the source to the ontology)
Learning Semantic Types

[Krishnamurthy et al., ESWC 2015]

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>H: 3.5 in, W: 2.5 in</td>
</tr>
<tr>
<td>H: 64 in, W: 51.5 in, D: .75 in</td>
</tr>
<tr>
<td>L: 57 in, center back: 23 in</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
## Learning Semantic Types

### CulturalHeritageObject

- **extent**

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>H: 3.5 in, W: 2.5 in</td>
</tr>
<tr>
<td>H: 64 in, W: 51.5 in, D: .75 in</td>
</tr>
<tr>
<td>L: 57 in, center back: 23 in</td>
</tr>
</tbody>
</table>

1. User specifies
2. System learns
Learning Semantic Types

CulturalHeritageObject

extent

**Dimensions**

- H: 3.5 in, W: 2.5 in
- H: 64 in, W: 51.5 in, D: .75 in
- L: 57 in, center back: 23 in
  ...

**Extent**

- 52.1 x 71.4 cm (20 1/2 x 28 1/8 in.)
- 9 3/4 x 7 9/16 in.
- H: 19 x W: 15 1/4 x D: 8 1/4 in.
  ...

...
Learning Semantic Types

CulturalHeritageObject

extent

Dimensions

H: 3.5 in, W: 2.5 in
H: 64 in, W: 51.5 in,
D: .75 in
L: 57 in, center
back: 23 in
...

CulturalHeritageObject

extent

Extent

52.1 x 71.4 cm (20
1/2 x 28 1/8 in.)
9 3/4 x 7 9/16 in.
H: 19 x W: 15 1/4 x
D: 8 1/4 in.
...

...
Requirements

• Learn from a small number of examples
• Work on both textual and numeric values
• Learn quickly and highly scalable to large number of semantic types
Approach for Textual Data

- **Document**: each column of data
- **Label**: each semantic type
- Use **Apache Lucene** to index the labeled documents
- Compute **TF/IDF vectors** for documents
- Compare documents using **Cosine Similarity** between TF/IDF vectors
Approach for Textual Data

Dimensions
- H: 3.5 in, W: 2.5 in
- H: 64 in, W: 51.5 in, D: .75 in
- L: 57 in, center back: 23 in

Extent
- 52.1 x 71.4 cm (20 1/2 x 28 1/8 in.)
- 9 3/4 x 7 9/16 in.
- H: 19 x W: 15 1/4 x D: 8 1/4 in.

Term: TF-IDF score
- h: 0.375
- w: 0.336
- in: 0.491
- centre: 0.241
- back: 0.301
- ...

Term: TF-IDF score
- h: 0.414
- w: 0.364
- d: 0.245
- cm: 0.354
- in: 0.395
- ...

TF-IDF Cosine Similarity

\[ tf(t, d) = \text{frequency}^{1/2} \]

\[ idf(t) = 1 + \log\left( \frac{\text{numDocs}}{\text{docFreq} + 1} \right) \]

\[ sim(q, d) = \frac{V(q) \cdot V(d)}{|V(q)| \cdot |V(d)|} \]
Approach for Numeric Data

• **Distribution** of values in different semantic types is different, e.g., temperature vs. population

• Use **Statistical Hypothesis Testing** to see which distribution fits best

• Welch’s T-test, Mann-Whitney U-test and **Kolmogorov-Smirnov Test**
Approach for Numeric Data

\[ D_{N_1,N_2} = \sup_x \left| F_{1,N_1}(x) - F_{2,N_2}(x) \right| \]
Similarity features

- Attribute names similarity
  - Jaccard
  - TF-IDF
- Value Similarity
  - Jaccard
- Distribution Similarity
  - Mann-Whitney test
  - Kolmogorov-Smirnov test
- Histogram Similarity
  - Mann-Whitney test
Training machine learning model
[Pham et al., ISWC 2016]
Predicting new attribute

Player
  └── birthName
    ├── player
    │   ├── Hazard, Eden
    │   │   └── height: 172
    │   └── Cahill, Gary
    │       └── height: 191
    └── Rooney, Wayne
        └── height: 176

Player
  └── height
    ├── ?
    └── Wayne Smith

Classifier
  └── jaccard: 0.4
  └── cosine: 0.5

Classifier
  └── jaccard: 0.05
  └── cosine: 0.1

Classifier
  └── P_{True} = 0.8

Classifier
  └── P_{True} = 0.1
Approach
[Knoblock et al, ESWC 2012]

Sample Data

Domain Ontology

Learn Semantic Types

Construct a Graph

Steiner Tree

Extract Relationships
Construct a Graph

Construct a graph from semantic types and ontology

<table>
<thead>
<tr>
<th></th>
<th>name</th>
<th>date</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>2</td>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
</tbody>
</table>
Construct a Graph

Construct a graph from semantic types and ontology
Inferring the Relationships

• Search for minimal explanation

• Steiner tree connecting semantic types over ontology graph
  • Given graph $G=(V,E)$, nodes $S \subset V$, cost $c: E \rightarrow \mathbb{R}$
  • Find a tree of $G$ that spans $S$ with minimal total cost
  • Unfortunately, NP-complete

• Approximation Algorithm [Kou et al., 1981]
  • Worst-case time complexity: $O(|V|^2|S|)$
  • Approximation Ratio: less than 2
Steiner Tree

Steiner nodes: \{V1, V2, V3, V4\}

1. construct the complete graph (Nodes: Steiner Nodes, Links Weights: shortest path from each pair in original G)

2. Compute MST

3. replace each link with the corresponding shortest path in original G

4. Compute MST

5. remove extra links until all leaves are Steiner nodes
Inferring the Relationships
Select minimal tree that connects all semantic types

- A customized Steiner tree algorithm
## Result in Karma

<table>
<thead>
<tr>
<th>name</th>
<th>birthDate</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
<tr>
<td>Richard Smith</td>
<td>Feb 1975</td>
<td>Los Angeles</td>
<td>CA</td>
<td>Apple</td>
</tr>
</tbody>
</table>
Refining the Model

Impose constraints on Steiner Tree Algorithm

- Change weight of selected links to $\epsilon$
- Add source and target of selected link to Steiner nodes
Final Semantic Model

<table>
<thead>
<tr>
<th>name</th>
<th>birthdate</th>
<th>city</th>
<th>state</th>
<th>workplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred Collins</td>
<td>Oct 1959</td>
<td>Seattle</td>
<td>WA</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Tina Peterson</td>
<td>May 1980</td>
<td>New York</td>
<td>NY</td>
<td>Google</td>
</tr>
<tr>
<td>Richard Smith</td>
<td>Feb 1975</td>
<td>Los Angeles</td>
<td>CA</td>
<td>Apple</td>
</tr>
</tbody>
</table>
Karma Learns the Source Models
Taheriyan et al., ISWC 2013, ICSC 2014

Sample Data

Domain Ontology

Learn Semantic Types

Construct a Graph

Generate Candidate Models

Rank Results

Known Semantic Models
Karma Use Cases
Source Mapping Phase

Mapping Phase
Source Mapping and Query Time

Mapping Phase

Domain Model

Samples of Source Data

Domain Expert

Source Mappings

Query Phase

Karma

Karma Runtime System

Data Warehousing

Virtual Integration

Query

Analyst
VIVO

- VIVO is a system to build researcher networks across institutions

- Used Karma to map the data about USC faculty to VIVO ontology and publish it as RDF

- VIVO ingest the RDF data

- Video
American Art Collaborative
[Knoblock et al., ISWC 2017]

- Used Karma to convert data of 13 American Art Museums to Linked Open Data

- Modeled according to CIDOC-CRM Ontology

- Linked the generated RDF to DBPedia and ULAN

- Video
Using Karma to map museum data to the CIDOC CRM ontology

https://www.youtube.com/watch?v=h3_yiBhAJlc
Discussion

• Automatically build rich semantic descriptions of data sources

• Exploit the background knowledge from (i) the domain ontology, and (ii) the known source models

• Semantic descriptions are the key ingredients to automate many tasks, e.g.,
  • Source Discovery
  • Data Integration
  • Service Composition
More Info

karma.isi.edu