Record Linkage

Craig Knoblock
University of Southern California

These slides are based in part on slides from
Sheila Tejada and Misha Bilenko
Record Linkage Problem

<table>
<thead>
<tr>
<th>Restaurant Name</th>
<th>Address</th>
<th>City</th>
<th>Phone</th>
<th>Cuisine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fenix</td>
<td>8358 Sunset Blvd. West</td>
<td>Hollywood</td>
<td>213/848-6677</td>
<td>American</td>
</tr>
<tr>
<td>Fenix at the Argyle</td>
<td>8358 Sunset Blvd.</td>
<td>W. Hollywood</td>
<td>213-848-6677</td>
<td>French (new)</td>
</tr>
</tbody>
</table>


• Task:
  identify syntactically different records that refer to the same entity

• Common sources of variation: database merges, typographic errors, abbreviations, extraction errors, OCR scanning errors, etc.
Outline

• Introduction
• Blocking
• Field Matching
• Record Matching
• Discussion
Integrating Restaurant Sources

Zagat’s Restaurant Guide Source

Department of Health Restaurant Rating Source

Information Mediator

Question: What is the Review and Rating for the Restaurant “Art’s Deli”?
Information Mediator

Extract web objects in the form of database records

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Deli</td>
<td>12224 Ventura Boulevard</td>
<td>818-756-4124</td>
</tr>
<tr>
<td>Teresa’s</td>
<td>80 Montague St.</td>
<td>718-520-2910</td>
</tr>
<tr>
<td>Steakhouse The</td>
<td>128 Fremont St.</td>
<td>702-382-1600</td>
</tr>
<tr>
<td>Les Celebrites</td>
<td>155 W. 58th St.</td>
<td>212-484-5113</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Delicatessen</td>
<td>12224 Ventura Blvd.</td>
<td>818/755-4100</td>
</tr>
<tr>
<td>Teresa’s</td>
<td>103 1st Ave. between 6th and 7th Sts.</td>
<td>212/228-0604</td>
</tr>
<tr>
<td>Binion’s Coffee Shop</td>
<td>128 Fremont St.</td>
<td>702/382-1600</td>
</tr>
<tr>
<td>Les Celebrites</td>
<td>5432 Sunset Blvd</td>
<td>212/484-5113</td>
</tr>
</tbody>
</table>
Application Dependent Mapping

Observations:

• Mapping objects can be application dependent
• Example:

<table>
<thead>
<tr>
<th>Steakhouse The</th>
<th>128 Fremont Street</th>
<th>702-382-1600</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binion's Coffee Shop</td>
<td>128 Fremont St.</td>
<td>702/382-1600</td>
</tr>
</tbody>
</table>

• The mapping is in the application, not the data
• User input is needed to increase accuracy of the mapping
General Approach to Record Linkage

1. Identification of candidate pairs
   • Comparing all possible record pairs would be computationally wasteful

2. Compute Field Similarity
   • String similarity between individual fields is computed

3. Compute Record Similarity
   • Field similarities are combined into a total record similarity estimate

4. Linkage/Merging
   • Records with similarity higher than a threshold are labeled as matches
   • Equivalence classes are found by transitive closure
Outline

• Introduction
• Blocking
• Field Matching
• Record Matching
• Discussion
Blocking

- Comparing all possible matches across two data sets would require $n^2$ comparisons
- On large datasets this is impractical and wasteful
- Instead, we compare only those that could possible be matched
- Also referred to as candidate generation
IR Approach to Blocking

• Construct an inverted index of all tokens in a document
  • Links the token to the documents in which it appears
  • Place each token in a hash table
• Apply transformations on the tokens to find closely related tokens
  • Transformations include equal, stemming, soundex, and other unary transformations
• Use a stop list to avoid common tokens
  • Tokens such as “the”, “s”, etc. would be on the stop list
Example: Partial Inverted Index for LA Department of Health

Document (Object) 5 Restaurant name: “Art’s Delicatessen”
Tokens: “Art”, “s”, “Delicatessen”

<table>
<thead>
<tr>
<th>Transformed Tokens</th>
<th>Transformations</th>
<th>Original Tokens</th>
<th>Object Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Art”</td>
<td>Equal</td>
<td>“Art”</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Stemming</td>
<td>“Arte”</td>
<td>57</td>
</tr>
<tr>
<td>“A630”</td>
<td>Stemming</td>
<td>“Art”</td>
<td>5</td>
</tr>
<tr>
<td>“s”</td>
<td>Soundex</td>
<td>“s”</td>
<td>5,6,9,71,79,97,111</td>
</tr>
<tr>
<td>“S000”</td>
<td>Equality</td>
<td>“s”</td>
<td>5,6,9,71,79,97,111</td>
</tr>
<tr>
<td>“Del”</td>
<td>Stemming</td>
<td>“Dell”</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>Stemming</td>
<td>“Deli”</td>
<td>7,93</td>
</tr>
<tr>
<td></td>
<td>Equality</td>
<td>“Del”</td>
<td>60</td>
</tr>
<tr>
<td>“Deli”</td>
<td>Equality</td>
<td>”Deli”</td>
<td>7,93</td>
</tr>
<tr>
<td>“D400”</td>
<td>Soundex</td>
<td>”Deli”</td>
<td>7,93</td>
</tr>
<tr>
<td></td>
<td>Soundex</td>
<td>“Dell”</td>
<td>57</td>
</tr>
<tr>
<td>“Delicatessen”</td>
<td>Equality</td>
<td>“Delicatessen”</td>
<td>5</td>
</tr>
<tr>
<td>“D423”</td>
<td>Soundex</td>
<td>“Delicatessen”</td>
<td>5</td>
</tr>
<tr>
<td>“Dell”</td>
<td>Equality</td>
<td>“Dell”</td>
<td>57</td>
</tr>
</tbody>
</table>
More on Blocking Later!
Outline

• Introduction
• Blocking
• Field Matching
• Record Matching
• Discussion
Field Matching Approaches

- Expert-system rules
  - Manually written
- Token similarity
  - Used in Whirl
- String similarity
  - Used in Marlin
- Domain-specific transformations
  - Used in Active Atlas
Token-based Metrics

• Any string can be treated as a bag of tokens.
  • “8358 Sunset Blvd” ▶ \{8358, Sunset, Blvd\}

• Each token corresponds to a dimension in Euclidean space; string similarity is the normalized dot product (cosine) in the vector space.

• Weighting tokens by Inverse Document Frequency (IDF) is a form of unsupervised string metric learning.
Token Similarity
[Cohen, 1998]

- Follows the same approach used by classical IR algorithms (including web search engines).
- First, “stemming” is applied to each entry.
  - E.g. “Joe’s Diner” -> “Joe [‘s] Diner”
- Then, entries are compared by counting the number of words in common.
- Note: Infrequent words weighted more heavily by TF/IDF metric = Term Frequency / Inverse Document Frequency
Sequence-based String Metrics: String Edit Distance [Levenshtein, 1966]

• Minimum number of character deletions, insertions, or substitutions needed to make two strings equivalent.
  • “misspell” to “mispell” is distance 1 (‘delete s’)
  • “misspell” to “mistell” is distance 2 (‘delete s’, ‘substitute p with t’ OR ‘substitute s with t’, ‘delete p’)
  • “misspell” to “misspelling” is distance 3 (‘insert i’, ‘insert n’, ‘insert g’)

• Can be computed efficiently using dynamic programming in $O(mn)$ time where $m$ and $n$ are the lengths of the two strings being compared.

• Unit cost is typically assigned to individual edit operations, but individual costs can be used.
String Edit Distance with Affine Gaps [Gotoh, 1982]

- Cost of gaps formed by contiguous deletions/insertions should be lower than the cost of multiple non-contiguous operators.
  - Distance from “misspell” to “misspelling” is <3.

- Affine model for gap cost: \( \text{cost}(\text{gap}) = s + e|\text{gap}|, \ e < s \)

- Edit distance with affine gaps is more flexible since it is less susceptible to sequences of insertions/deletions that are frequent in natural language text (e.g. ‘Street’ vs. ‘Str’).
Learnable Edit Distance with Affine Gaps

• Motivation:

  Significance of edit operations depends on a particular domain
  • Substitute ‘/’ with ‘-‘ insignificant for phone numbers.
  • Delete ‘Q’ significant for names.
  • Gap start/extension costs vary: sequence deletion is common for addresses (‘Street’ ▶ ‘Str’), uncommon for zip codes.

• Using individual weights for edit operations, as well as learning gap operation costs allows adapting to a particular field domain.

• [Ristad & Yianilos, ‘97] proposed a one-state generative model for regular edit distance.
Learnable Edit Distance with Affine Gaps – the Generative Model

- Matching/substituted pairs of characters are generated in state $M$.
- Deleted/inserted characters that form gaps are generated in states $D$ and $I$.
- Special termination state “#” ends the alignment of two strings.
- Similar to pairwise alignment HMMs used in bioinformatics [Durbin et al. 1998]
Learnable Edit Distance with Affine Gaps: Training

- Given a corpus of *matched* string pairs, the model is trained using Expectation-Maximization.
- The model parameters take on values that result in high probability of producing duplicate strings.
  - Frequent edit operations and typos have *high* probability.
  - Rare edit operations have *low* probability.
  - Gap parameters take on values that are optimal for duplicate strings in the training corpus.
- Once trained, distance between any two strings is estimated as *the posterior probability of generating the most likely alignment between the strings as a sequence of edit operations*.
- Distance computation is performed in a simple dynamic programming algorithm.
Learning Transformation Weights

- Learn general transformations to recognize related objects

Zagat’s  Transformations  Dept of Health

Art’s Deli  Prefix  Art’s Delicatessen
California Pizza Kitchen  Acronym  CPK
Philippe The Original  Stemming  Philippe’s The Original
Transformation Weights

- Transformations can be more appropriate for a specific application domain
  - Restaurants, Companies or Airports

- Or for different attributes within an application domain
  - Acronym more appropriate for the attribute Restaurant Name than for the Phone attribute

- Learn likelihood that if transformation is applied then the objects are mapped

Transformation Weight = P(mapped | transformation)
Types of Transformations

Unary Transformations
- Equality (Exact match)
- Stemming
- Soundex (e.g. “Celebrites” => “C453”)
- Abbreviation (e.g. “3rd” => “third”)

Binary Transformations
- Initial
- Prefix (e.g. “Deli” & “Delicatessen”)
- Suffix
- Substring
- Acronym (e.g. “California Pizza Kitchen” & “CPK”)
- Drop Word
Applying Unary Transformations

Employs Information Retrieval Techniques
- One set of attribute values broken into words or tokens
  - “Art” “s” “Delicatessen”
- Apply Type I transformations to tokens
  - “Art” “A630” “s” “S000” “Delicatessen” “D423”
- Enter tokens into inverted index
- Tokens from second set used to query the index
  - Transformed query set: “Art” “A630” “s” “S000” “Deli” “Del” “D400”
Applying Binary Transformations

- Binary transformations improve measurement of similarity
Calculate Transformation Weights

\[
P(mapped \mid transformation) = \frac{P(transformation \mid mapped) \cdot P(mapped)}{P(transformation)}
\]

<table>
<thead>
<tr>
<th>Examples</th>
<th>Classification</th>
<th>Labeled by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Deli, Art’s Delicatessen</td>
<td>Mapped</td>
<td>Learner</td>
</tr>
<tr>
<td>CPK, California Pizza Kitchen</td>
<td>Mapped</td>
<td>User</td>
</tr>
<tr>
<td>Ca’Brea, La Brea Bakery</td>
<td>Not Mapped</td>
<td>Learner</td>
</tr>
</tbody>
</table>
Computing Textual Similarity

Zagat’s Restaurant Objects

Z1, Z2, Z3

Department of Health Objects

D1, D2, D3

W

S\text{\_name} S\text{\_street} S\text{\_phone}

• Candidate Generator returns sets of similarity scores

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>.9</td>
<td>.79</td>
<td>.4</td>
</tr>
<tr>
<td>.17</td>
<td>.3</td>
<td>.74</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Outline

• Introduction
• Candidate Generation
• Field Matching
• Record Matching
• Discussion
Combining String Similarity Across Fields

- Some fields are more indicative of record similarity than others:
  - For addresses, *street address* similarity is more important than *city* similarity.
  - For bibliographic citations, *author* or *title* similarity are more important than *venue* (i.e. conference or journal name) similarity.

- Field similarities should be weighted when combined to determine record similarity.

- Weights can be learned using a learning algorithm [Cohen & Richman ‘02], [Sarawagi & Bhamidipaty ‘02], [Tejada et al. ‘02].
Record Matching Approaches

- Learning Decision Trees
- Support Vector Machines (SVM)
- Unsupervised Learning
Learning Mapping Rules with Decision Trees

- Learning important attributes for determining a mapping

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zagat’s</td>
<td>Art’s Deli 12224 Ventura Blvd</td>
<td>818-756-4124</td>
</tr>
<tr>
<td>Dept of Health</td>
<td>Art’s Delicatessen 12224 Ventura Blvd.</td>
<td>818/755-4100</td>
</tr>
</tbody>
</table>
Learning Mapping Rules with Decision Trees

Mapping rules:

Name > .9 & Street > .87 => mapped

Name > .95 & Phone > .96 => mapped

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Deli</td>
<td>12224 Ventura Boulevard</td>
<td>818-756-4124</td>
</tr>
<tr>
<td>Teresa’s</td>
<td>80 Montague St.</td>
<td>718-520-2910</td>
</tr>
<tr>
<td>Steakhouse The</td>
<td>128 Fremont St.</td>
<td>702-382-1600</td>
</tr>
<tr>
<td>Les Celebrites</td>
<td>155 W. 58th St.</td>
<td>212-484-5113</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Delicatessen</td>
<td>12224 Ventura Blvd.</td>
<td>818/755-4100</td>
</tr>
<tr>
<td>Teresa’s</td>
<td>103 1st Ave. between 6th and 7th Sts.</td>
<td>212/228-0604</td>
</tr>
<tr>
<td>Binion’s Coffee Shop</td>
<td>128 Fremont St.</td>
<td>702/382-1600</td>
</tr>
<tr>
<td>Les Celebrites</td>
<td>160 Central Park S</td>
<td>212/484-5113</td>
</tr>
</tbody>
</table>
Learning Mapping Rules

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>.967</td>
<td>.973</td>
<td>.3</td>
</tr>
<tr>
<td>.17</td>
<td>.3</td>
<td>.74</td>
</tr>
<tr>
<td>.8</td>
<td>.542</td>
<td>.49</td>
</tr>
<tr>
<td>.95</td>
<td>.97</td>
<td>.67</td>
</tr>
</tbody>
</table>

Name > .8 & Street > .79 => mapped
Name > .89 => mapped
Street < .57 => not mapped
Mapping Rule Learner with Active Learning

Choose initial examples

Generate committee of learners

Learn Rules
Classify Examples
Votes

Learn Rules
Classify Examples
Votes

Learn Rules
Classify Examples
Votes

Choose Example

Set of Mapped Objects

Label

USER

Label
Committee Disagreement

- Chooses an example based on the disagreement of the query committee

<table>
<thead>
<tr>
<th>Examples</th>
<th>Committee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Deli, Art’s Delicatessen</td>
<td>Yes</td>
</tr>
<tr>
<td>CPK, California Pizza Kitchen</td>
<td>Yes</td>
</tr>
<tr>
<td>Ca’Brea, La Brea Bakery</td>
<td>No</td>
</tr>
</tbody>
</table>

- In this case CPK, California Pizza Kitchen is the most informative example based on disagreement
SVM Learned Record Similarity

- String similarities for each field are used as components of a feature vector for a pair of records.

- SVM is trained on labeled feature vectors to discriminate duplicate from non-duplicate pairs.

- Record similarity is based on the distance of the feature vector from the separating hyperplane.
Learning Record Similarity (cont.)

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>City</th>
<th>Cuisine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fenix</td>
<td>8358 Sunset Blvd. West</td>
<td>Hollywood</td>
<td>American</td>
</tr>
<tr>
<td>Fenix at the Argyle</td>
<td>8358 Sunset Blvd.</td>
<td>W. Hollywood</td>
<td>French (new)</td>
</tr>
</tbody>
</table>

Learned distance measure

Feature vector

\[
\begin{bmatrix}
  d_{1n} & d_{2n} & d_{1a} & d_{2a} & d_{1c} & d_{2c} & d_{1cu} & d_{2cu}
\end{bmatrix}
\]

SVM

Distance

Duplicate records

Non-duplicate records
Learnable Vector-space Similarity

x: “3130 Piedmont Road”
y: “3130 Piedmont Rd. NE”

Each string is converted to vector-space representation

\[
\begin{pmatrix}
x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\
\end{pmatrix}
\quad
\begin{pmatrix}
y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\
\end{pmatrix}
\]

The pair vector is created

\[
\begin{pmatrix}
x_1y_1 \\ x_2y_2 \\ x_3y_3 \\ x_4y_4 \\ x_5y_5 \\
\end{pmatrix}
\]

The pair vector is classified as “similar” or “dissimilar”

\[
S \propto f(p(x,y))
\]

Similarity between strings is obtained from the SVM output

\[
Sim(x, y) \propto f(p^{(x,y)})
\]
Unsupervised Record Linkage

- Idea: Analyze data and automatically cluster pairs into three groups:
  - Let $R = P(\text{obs} | \text{Same}) / P(\text{obs} | \text{Different})$
  - Matched if $R > \text{threshold} \ T_U$
  - Unmatched if $R < \text{threshold} \ T_L$
  - Ambiguous if $T_L < R < T_U$

- This model for computing decision rules was introduced by Felligi & Sunter in 1969

- Particularly useful for statistically linking large sets of data, e.g., by US Census Bureau
Unsupervised Record Linkage (cont.)

• Winkler (1998) used EM algorithm to estimate $P(\text{obs} | \text{Same})$ and $P(\text{obs} | \text{Different})$
• EM computes the *maximum likelihood estimate*. The algorithm iteratively determines the parameters most likely to generate the observed data.
• Additional mathematical techniques must be used to adjust for “relative frequencies”, i.e. last name of “Smith” is much more frequent than “Knoblock”.

---

Craig Knoblock  
University of Southern California  
41
Outline

• Introduction
• Candidate Generation
• Field Matching
• Record Matching
• Discussion
Enforcing One-to-One Relationship

- Viewed as weighted bipartite matching problem

Zagat’s

(Name, Street, City)
(Art’s Deli, 1745 Ventura Boulevard, Encino)
(Citrus, 267 Citrus Ave., LA)
(Spago, 456 Sunset Bl., LA)
(Z1, Z2, Z3)

Dept of Health

(Name, Street, City)
(Art’s Delicatessen, 1745 Ventura Blvd, Encino)
(Ca’ Brea, 6743 La Brea Ave., LA)
(Patina, 342 Melrose Ave., LA)
(D1, D2, D3)

Given weights W, matching method determines mostly likely Matching Assignment
Related Work

- Record linkage [Newcombe et al. ’59; Fellegi & Sunter ’69; Winkler ’94, ’99, ‘02]
- Database hardening [Cohen et al. ’00]
- Merge/purge [Hernandez & Stolfo ’95]
- Field matching [Monge & Elkan ’96]
- Data cleansing [Lee et al. ’99]
- Name matching [Cohen & Richman ’01, Cohen et al. ’03]
- De-duplication [Sarawagi & Bhamidipaty ’02]
- Object identification [Tejada et al. ’01, ’02]
- Fuzzy duplicate elimination [Ananthakrishna et al. ’02]
- Identity uncertainty [Pasula et. al. ’02, McCallum & Wellner ’03]
- Object consolidation [Michalowski et al. ’03]
Conclusions

• Technical choices in record linkage:
  • Approach to blocking
  • Approach to field matching
  • Approach to record matching

• Learning approaches have the advantage of being able to
  • Adapt to specific application domains
  • Learn which fields are important
  • Learn the most appropriate transformations

• Optimal classifier choice is sensitive to the domain and the amount of available training data.