Today,

• Duplicate Detection Algorithm: clustering algorithm
• Evaluating Detection Success

Duplicate Detection Algorithms:
Clustering algorithms

• Pairwise comparison algorithms
• Algorithms for data with complex relationships
• Clustering algorithms

Goal

• Finding duplicate clusters
  – Partition a set of candidates C (of given type T) into set of candidates where each set represents a different real-world object and all candidates within a set are different representations of the same real-world object.

The Transitive Closure

• Definition: Let R be a binary relation on a set A. The transitive closure of R is the binary relation $R'$ on A satisfying the following three properties:
  1. $R'$ is transitive
  2. $R'$ is a subset of $R$
  3. If $S$ is any other transitive relation that contains $R$, then $S$ contains $R'$.
• In other words, the transitive closure of $R$ is the smallest transitive relation containing $R$.

Example of the Transitive Closure

• Given the relation $R$ on {1,2,3,4},

1 – 2

4 – 3

• its transitive closure is:

1 – 2

4 – 3

The relationship “is-duplicate-of”

• Is transitive.
  – Candidate 3 is a duplicate of candidate 5
  – Candidate 3 is a duplicate of candidate 6
  – Candidate 5 and 6 are duplicate
Example: Media database

- Media database that stores music tracks and an artist
- Duplicates are \{(1,2), (3,4), and (9,10,11)\}

<table>
<thead>
<tr>
<th>ID</th>
<th>Artist</th>
<th>Track</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tori Amos</td>
<td>Beekeeper</td>
</tr>
<tr>
<td>2</td>
<td>Amos, Tori</td>
<td>Beekeeper</td>
</tr>
<tr>
<td>3</td>
<td>Beethoven</td>
<td>Symphony No.5</td>
</tr>
<tr>
<td>4</td>
<td>Ludwig van Beethoven</td>
<td>5th Symphony</td>
</tr>
<tr>
<td>5</td>
<td>Beethoven</td>
<td>Symphony No.1</td>
</tr>
<tr>
<td>6</td>
<td>Beethoven</td>
<td>Symphony No.2</td>
</tr>
<tr>
<td>7</td>
<td>Beethoven</td>
<td>Symphony No.3</td>
</tr>
<tr>
<td>8</td>
<td>Schubert</td>
<td>Symphony No.1</td>
</tr>
<tr>
<td>9</td>
<td>AC/DC</td>
<td>Are you ready</td>
</tr>
<tr>
<td>10</td>
<td>AC/DC</td>
<td>Are you ready</td>
</tr>
<tr>
<td>11</td>
<td>AC/DC</td>
<td>Are you ready</td>
</tr>
<tr>
<td>12</td>
<td>Bob Dylan</td>
<td>Are you ready</td>
</tr>
<tr>
<td>13</td>
<td>Michael</td>
<td>Thriller</td>
</tr>
</tbody>
</table>

Duplicate Pair Graph

- Using a similarity-based duplicate classification
  - \{(1,2), (3,4), (3,5), (5,6), (6,7), (5,8), (9,10), (9,11), (10,11), (9,12), (10,12)\}

Partitioning based on the connected components

- Based on the observation that the relationship “is-duplicate-of” is transitive
  - Track candidate 3 is duplicate of both 5 and 4, 4 and 5 are duplicate as well.

Edge Removal

- Revoke duplicate classifications to obtain more connected components
  - Recursively remove the edge with the lowest similarity until a satisfactory result is obtained.

- Satisfactory result
  - We do not have transitivity paths longer than a certain number of steps
  - Lowest similarity between any two candidates should not fall under a given threshold
Partitioning based on the connected components: Removing Edges

- Determine centers within a connected component
- Require that a candidate is part of the cluster of the closest center

1. Sort the edges of the duplicate pair graph in descending order of their similarity.
2. Scan the sorted set of edges and a candidate c that occurs first becomes a center.
3. All edges including the candidate connected to c becomes part of the cluster.

Make all candidates as part of a cluster

- Merging clusters (threshold = 0.7)

Evaluating Detection Success
Evaluating Detection Success

• False positives
  – Candidate pairs that are declared to be duplicates may not be duplicates
• False negatives
  – Candidate pairs that were not declared to be duplicates while they are

Precision and Recall

Precision = \frac{|true\ -\ positives|}{|true\ -\ positives| + |false\ -\ positives|}
\quad = \frac{|true\ -\ positives|}{|declared\ _duplicates|}

Recall = \frac{|true\ -\ positives|}{|true\ -\ positives| + |false\ -\ negatives|}
\quad = \frac{|true\ -\ positives|}{|true\ _duplicates|}

Optimizing Precision and Recall

• Optimizing Precision
  – Play safe!
  – Return very few (or no) duplicates
  – Minimum false positives
• Optimizing Recall
  – Maximum recall: n^2 candidate pairs

• There are tradeoffs between optimizing precision and recall.

Arithmetic Mean vs. Harmonic Mean

Finding Tradeoff between precision and recall

• F-measure
  – Harmonic mean of precision and recall:
  \quad F - measure = \frac{2 \times recall \times precision}{recall + precision}
Recall-Precision Diagram

Recall-Precision-F-measure Diagram

From Creating probabilistic databases from duplicated data
Oleksiy Vaznyak, Renée J. Miller (2005)