Programmable Similarity for Record Matching

Arvind Arasu
Database Group, Microsoft Research

Joint work with Data Cleaning Research Team @ Microsoft Research
Record Matching

- Automate answering:
  - Do two records (texts) correspond to the same entity?

- Search and analysis applications:
  - Online map-services
    - Address matching
  - Citations: Citeseer, Google Scholar, Bing Academic
    - Citation matching
  - Comparative shopping sites
    - Product matching
Example: Citations


by Sergey Brin, Lawrence Page

Venue: COMPUTER NETWORKS AND ISDN SYSTEMS

Tools

Sorted by:
Citation Count

1. Authoritative Sources in a Hyperlinked Environment
   by Jon M. Kleinberg - JOURNAL OF THE ACM, 1999
   "...The network structure of a hyperlinked environment can be a rich source of information about the content of the environment, provided we have effective means for understanding it. We develop a set of algorithmic tools for extracting information from the link structure of such environments, and report..."
   Abstract - Cited by 2003 (9 self) - Add to MetaCart

2. The PageRank Citation Ranking: Bringing Order to the Web
   by Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd - Stanford InfoLab, 1999
   "...The importance of a Web page is an inherently subjective matter, which depends on the readers' interests, knowledge and attitudes. But there is still much that can be said objectively about the relative importance of Web pages. This paper describes PageRank, a method for rating Web pages objectively..."
   Abstract - Cited by 1471 (1 self) - Add to MetaCart

3. The structure and function of complex networks
   "...Inspired by empirical studies of networked systems such as the Internet, social networks, and biological networks, researchers have in recent years developed a variety of techniques and models to help us understand and predict the behavior of these systems. Here we review developments in this field..."
   Abstract - Cited by 973 (3 self) - Add to MetaCart

4. Video google: A text retrieval approach to object matching in videos
   by Josef Sivic, Andrew Zisserman - In Proc. CVG, 2003
   "...We describe an approach to object and scene retrieval which searches for and localizes all the occurrences of a user outlined object in a video. The object is represented by a set of viewpoint invariant region descriptions so that recognition can proceed successfully despite changes in viewpoint, ill..."
   Abstract - Cited by 432 (24 self) - Add to MetaCart
Most cited computer science authors?

Most Cited Computer Science Authors

This is generated from documents in the CiteSeer database as of September 18, 2011. An entry may correspond to multiple authors (e.g. J. Smith). This list is automatically generated and may contain errors. Citation counts may differ from search results because this list is generated in batch mode whereas the database is continually updated.

1. D. Johnson
   31772
2. J. Smith
   22791
3. Y. Wang
   21674
4. J. Lee
   20341
5. A. Gupta
   19642
6. L. Zhang
   19584
7. J. Wang
   18851
8. R. Rivest
   18829
Address Matching

Results are for prairie crossing dr w chicago il 60185. Get results for prairie crossing dr w chicago il 60185.

Popular categories
Restaurants
Bars, Grills & Pubs
Malls & Shopping Centers
More

Explore user-contributed places

NEARBY
Sushi Yama
Reviews (7)
Product Matching

Garmin Nuvi 350 - GPS receiver
The sleek, portable Nuvi 350 is a GPS navigator, traveler's reference reference and digital entertainment system, all in one. It is your pocket-sized personal travel assistant ready for... more.

5 stars ★★★★★ User reviews (645)
Expert reviews (1)

Garmin Nuvi 350 GPS receiver
Introducing the Nuvi: A versatile travel assistant that's approximately approximately the size of a deck of playing cards. The Nuvi is a portable GPS navigator, traveler's reference, and... more.

5 stars ★★★★★ User reviews (293)
Expert reviews (1)

Garmin Nuvi 360 - GPS receiver, Automotive, 320 x 240, WAAS 12
Navigator, Translator, Entertainer, Tour Guide. Garmin's Nuvi 360 is one versatile little GPS. This pocket-sized Personal Travel Assistant now comes with hands-free Bluetooth... more.

5 stars ★★★★★ User reviews (153)
Record Matching: State-of-the-art

- Problem Characteristics:
  - AI-complete?
  - Classification problem

- Standard approach:
  - Textual similarity as a signal
Textual Similarity for Matching


Textual Similarity for Matching


Record Matching: State-of-the-art

- Problem Characteristics:
  - AI-complete?
  - Classification problem

- Standard approach:
  - Textual similarity as a signal
  - Details:
    - Combining similarities from different columns (signals)
    - Learning approaches
    - Performance optimizations
  - Focus of this talk: How to measure textual (string) similarity?
Overview

- Introduction

- Textual Similarity
  - Limitations of current similarity functions

- Programmable Similarity
  - Semantics
  - Usability
  - Performance

- Conclusion
Textual Similarity

- String Similarity Function:
  \[ Sim(string, string) \rightarrow \text{numeric value} \]

- A “good” similarity function:
  - Strings representing the same concept \( \Rightarrow \) high similarity
  - Strings representing different concepts \( \Rightarrow \) low similarity
Edit Distance

- EditDistance (s1, s2): Minimum number of edits to transform s1 to s2

- Edit:
  - Insert a character
  - Delete a character
  - Substitute a character

- Note: EditDistance(s1, s2) = EditDistance (s2, s1)

- “distance” opposite of “similarity”
Edit Distance

EditDistance ("Seattle", "Siatle") = 2

EditDistance ("Seattle", "Redmond") = 6
Edit Distance Limitations

148th Ave NE, Redmond, WA

EditDist = 1

140th Ave NE, Redmond, WA

University Avenue, Seattle, WA

EditDist = 3

University Ave, Seattle, WA
Jaccard Similarity

- Statistical measure
- Originally defined over sets
- String = set of words

\[ Jaccard(s1, s2) = \frac{|s1 \cap s2|}{|s1 \cup s2|} \]

- Range of values: [0, 1]
Jaccard Similarity

\[ Jaccard = \frac{a}{a + b} \approx 0.66 \]

148th Ave NE, Redmond, WA

140th Ave NE, Redmond, WA
Weighted Jaccard Similarity

Weight Function = \( wt: Elements \rightarrow \mathbb{R}^+ \)

\[
WtJaccard(s_1, s_2) = \frac{wt(s_1 \cap s_2)}{wt(s_1 \cup s_2)}
\]

\[
wt(s) = \sum_{e \in s} wt(e)
\]
List of other Similarity Functions

- Affine edit distance
- Cosine similarity
- Hamming distance
- Generalized edit distance
- Jaro distance
- Monge-Elkan distance
- Q-gram
- Smith-Waterman distance
- Soundex
- TF/IDF
- ... many more
Jaro-Winkler distance

Definition

The Jaro distance $d_j$ of two given strings $s_1$ and $s_2$ is

$$d_j = \frac{1}{3} \left( \frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m - t}{m} \right)$$

where:

- $m$ is the number of matching characters (see below);
- $t$ is half the number of transpositions (see below).

Two characters from $s_1$ and $s_2$ respectively, are considered matching only if they are not farther than $\left\lfloor \frac{\max(|s_1|, |s_2|)}{2} \right\rfloor - 1$.

Each character of $s_1$ is compared with all its matching characters in $s_2$. The number of matching (but different sequence order) characters divided by the numeric value '2' defines the number of transpositions. For example, in comparing CRATE with TRACE, only 'R' 'A' 'E' are the matching characters, i.e., $m=3$. Although 'C', 'T' appear in both strings, they are farther than 1.5, i.e., $(5/2)-1=1.5$. Therefore, $t=0$. In DwAyNE versus DuANE the matching letters are already in the same order D-A-N-E, so no transpositions are needed.

Jaro–Winkler distance uses a prefix scale $p$ which gives more favourable ratings to strings that match from the beginning for a set prefix length $\ell$. Given two strings $s_1$ and $s_2$, their Jaro–Winkler distance $d_w$ is:

$$d_w = d_j + (\ell p(1 - d_j))$$

where:

- $d_j$ is the Jaro distance for strings $s_1$ and $s_2$
- $\ell$ is the length of common prefix at the start of the string up to a maximum of 4 characters
- $p$ is a constant scaling factor for how much the score is adjusted upwards for having common prefixes. $p$ should not exceed 0.25, otherwise the distance can become larger than 1. The standard value for this constant in Winkler's work is $p = 0.1$

Although often referred to as a distance metric, the Jaro–Winkler distance is actually not a metric in the mathematical sense of that term.
List of other Similarity Functions

- Affine edit distance
- Cosine similarity
- Hamming distance
- Generalized edit distance
- Jaro distance
- Monge-Elkan distance
- Q-gram
- Smith-Waterman distance
- Soundex
- TF/IDF
- … many more

- Limitation: “variations” syntactic & predefined
Complex Variations


- Synonyms
- Abbreviations
- Missing/additional information
Our approach

- Programmable Similarity:
  - Simple similarity function (Jaccard)
  - Variations as explicit input ("program")
Programmable Similarity

Transformation Rules:
- St → Street
- St → Saint
- Ave → Avenue
- 4th → Fourth

Programmable Similarity Framework

Jaccard Similarity
Programmable Similarity

St is an alternate representation of Street

Transformation Rules

St → Street
St → Saint
Ave → Avenue
4th → Fourth

Programmable Similarity Framework

Jaccard Similarity
Programmable Similarity

Transformation Rules

St → Street
St → Saint
Ave → Avenue
4th → Fourth

Programmable Similarity Framework

Jaccard Similarity

4th Ave Seattle

Fourth Avenue Seattle

1.0
Programmable Similarity

Transformation Rules:
- Jeff → Jeffrey
- Mike → Michael
- Amy → Amelia
- Bob → Robert

Programmable Similarity Framework

Jaccard Similarity

4th Ave Seattle

0.166

Fourth Avenue Seattle
Programmable Similarity

Transformation Rules

Jeff → Jeffrey
Mike → Michael
Amy → Amelia
Bob → Robert

Programmable Similarity Framework

Jaccard Similarity

0.91

Jeff Ullman

Jeffrey D Ullman
Programmable Similarity: Semantics

Transformation Rules

Jeff → Jeffrey
J → Jeffrey
J → John
J → Jack

Programmable Similarity Framework

J Ullman
Jeffrey Ullman
John Ullman
Jack Ullman

J Ullman

Jeff D Ullman
Jeffrey D Ullman

Jaccard Similarity

0.91
Programmable Similarity: Semantics

Transformation Rules

Jeff → Jeffrey
J → Jeffrey
J → John
J → Jack

Programmable Similarity Framework

Jaccard Similarity

J Ullman
Jeffrey Ullman
John Ullman
Jack Ullman

Jeff D Ullman
Jeffrey D Ullman

0.91
Programmable Similarity: Semantics

Transformation Rules

Jeff → Jeffrey
J → Jeffrey
J → John
J → Jack

0.91

Programmable Similarity Framework

Jaccard Similarity

Jeff D Ullman

Jeffrey D Ullman

Jeffrey Ullman

John Ullman

Jack Ullman

J Ullman

Jeff D Ullman

Jeffrey Ullman
Programmable Similarity: Semantics

\[ \text{ProgSim}(s1, s2): \ \text{Max}_{j,k} \ Jaccard(s1j, s2k) \]
Overview

- Introduction
- Textual Similarity
  - Limitations of current similarity functions
- Programmable Similarity
  - Semantics
  - Usability
  - Performance
- Conclusion
Nonsensical Variations?

Transformation Rules

- St → Street
- St → Saint
- Ave → Avenue
- 4th → Fourth

Programmable Similarity Framework

Jaccard Similarity

Transformation Examples:
- 4th St Seattle
- 4th Street Seattle
- 4th Saint Seattle
- Fourth Street Seattle

4th St Seattle

Fourth Street Seattle
Nonsensical Variations?

Transformation Rules
- St → Street
- St → Saint
- Ave → Avenue
- 4th → Fourth

Programmable Similarity Framework

Jaccard Similarity

Overall similarity not affected
Programmable Similarity

Progsim(s1, s2): Max_{j,k} Jaccard(s1j, s2k)
Similarity for Citations


S → Sergey
L → Larry
7th → Seventh
Proc → Proceedings of the
Similarity for Citations


How to anticipate and enumerate all these variations?
Example Rule Class: Edits

\[ \{ w' \rightarrow w \mid w \in \text{Dictionary of Words} \land \text{EditDistance}(w', w) \leq k \} \]

- Stanf0rd $\rightarrow$ Stanford
- Stnford $\rightarrow$ Stanford
- Stanfrd $\rightarrow$ Stanford
- Berkley $\rightarrow$ Berkeley

... ... ...

Too Large?

Dictionary of Cities

Stanford
Berkeley
...

...
Dynamic Edit Rules

Transformation Rules

Univ → University

Programmable Similarity Framework

Jaccard Similarity

Stanford Univ

Stanford University
Dynamic Edit Rules

Transformation Rules

- Univ → University
- Stanfrd → Stanford

Programmable Similarity Framework

Jaccard Similarity

Stanfrd Univ

Stanford University
Example Rule Class: First Name Initials

\{ l \rightarrow w \mid w \in \text{Dictionary of FirstNames} \land w = lu \land l \in \text{letters} \}\n
H \rightarrow Hector
J \rightarrow Jeffrey
J \rightarrow Jennifer
...

Dictionary of First Names
Hector
Jeffrey
Jennifer
...

43
Example Rule Class: Number related

<table>
<thead>
<tr>
<th>1</th>
<th>1st</th>
</tr>
</thead>
<tbody>
<tr>
<td>→</td>
<td>→</td>
</tr>
<tr>
<td>One</td>
<td>First</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2</th>
<th>2nd</th>
</tr>
</thead>
<tbody>
<tr>
<td>→</td>
<td>→</td>
</tr>
<tr>
<td>Two</td>
<td>Second</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>→</td>
<td>→</td>
</tr>
<tr>
<td>Three</td>
<td>Third</td>
</tr>
</tbody>
</table>

... ... ...

Easily generated programmatically
Similarity for Citations


How to anticipate and enumerate all these variations?
Web as a source of rules

<table>
<thead>
<tr>
<th>Web of Things (WoT)</th>
<th>Workshop on Trustworthy Elections (WOTE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSC</td>
<td>Workshop on Privacy in the Electronic Society (WPES)</td>
</tr>
<tr>
<td>WPMC</td>
<td>Wireless Personal Multimedia Communications (WPNC)</td>
</tr>
<tr>
<td>WWV</td>
<td>Workshop on Automated Specification and Verification of Web Sites (WWW)</td>
</tr>
<tr>
<td>WWW</td>
<td>International World Wide Web Conferences (WWW)</td>
</tr>
</tbody>
</table>

WoT → Web of Things

... → ...

WWV → Workshop on Automated Specification and ...

WWW → International World Wide Web Conferences
How to anticipate and enumerate all these variations?
Learning from Examples

60460 Hwy 50 Olathe CO

60460 Highway 50 Olathe CO
Learning from Examples

60460 Hwy 50 Olathe CO

60460 Highway 50 Olathe CO

Hwy → Highway

Alignment

A matching involving (hyper-)edges
Problem Formulation

Output $k$ transformations that maximize alignment of input matching strings.

Comments:

• As we increase $k$ correct transformations start appearing before incorrect ones.
• There is a greedy $\frac{1}{2} \left(1 - \frac{1}{e^2}\right) = 0.43$ approximation algorithm
• Connections to Machine Translation
  • Thanks: Dr. Fernando Pereira
Source of Rules (Summary)

- Manual
- Web
- Programmatic
- Learning
Overview

- Introduction
- Textual Similarity
  - Limitations of current similarity functions
- Programmable Similarity
  - Semantics
  - Usability
  - Performance
- Conclusion
Computing Similarity

- Compute similarity of string $s_1$ and $s_2$ under transformations $R$
- Undecidable in general 😞
- Engineering simplification
  - Only one “level” of derivation while applying transformations
Non-recursive derivation

Transformation Rules

A → U
B → V
C → W
U → Y

Programmable Similarity Framework

Basic Similarity

A B C D

A C X Y
A W X Y
Non-recursive derivation

Transformation Rules

A \rightarrow U
B \rightarrow V
C \rightarrow W
U \rightarrow Y

Programmable Similarity Framework

Basic Similarity

ACXY
AWXY
Non-recursive derivation

Transformation Rules

A B → U
B → V
C → W
U → Y

Programmable Similarity Framework

Basic Similarity

A B C D

U C D
A V C D
A B W D
U W D
A B C D
A V W D

A C X Y
A W X Y
Computing Similarity

- Compute similarity of string \( s_1 \) and \( s_2 \) under transformations \( R \)

- Undecidable in general 😞

- Engineering simplification
  - Only one “level” of recursion while applying transformations
Running Example

Tokens/Words: A, B, C, …, Z, a, b, …, z

Transformation Rules:

A → a
B → b
C → c

... ... ...

Z → z
Computing Similarity
Computing Similarity

\[ \left\{ \begin{array}{c}
P Q R S T \\
p Q R S T \\
P q R S T \\
P Q r S T \\
\vdots \\
p q r s t \\
\end{array} \right. \quad \quad 4 \mid 6 = 2 \mid 3 \quad \quad \left\{ \begin{array}{c}
P q T S t \\
p q T S t \\
P q t S t \\
\vdots \\
p q t s t \\
\end{array} \right. \]

Stanford Infoseminar 1/27/2012
Computing Similarity

- NP-Hard in general
- Polynomial for *unit rules*
  - Reduce to maximum bipartite matching
  - Note: Works only for Jaccard variants

Unit rule: $A \rightarrow a$
Multi rule: $A \rightarrow a$
Multi rule: $A \rightarrow a \ b$
Reduction to Matching

\[
P \quad Q \quad R \quad S \quad T
\]

\[
P \quad P
Q \quad q
R \quad T
S \quad S
T \quad t
\]

\[
P \quad q \quad T \quad S \quad t
\]
Reduction to Matching
Reduction to Matching
Reduction to Matching

Max Intersection = Max Matching = 4

Max Jaccard = Max Intersection / (10 – Max Intersection) = 4/6 = 2/3
Computing Similarity

- NP-Hard in general
- Polynomial for *unit rules*
  - Reduce to maximum bipartite matching
- General Heuristic
  - Enumerate all variations due to multi-rules
  - Use polynomial algorithm for each pair of variations
  - Works well in practice
    - Unit rules more common
    - Multi rules produce fewer variations
Record Matching: Practical Considerations

- **Index Setting:**
  - **Input:** Relation \( S \) (to index) and a single record \( r \)
  - **Output:** All records of \( S \) with similarity \( \geq \theta \) with \( r \)

- **Join Setting (Similarity Join):**
  - **Input:** Two relations \( R \) and \( S \)
  - **Output:** All pairs of records from \( R \) and \( S \) with similarity \( \geq \theta \)

*Similarity in presence of transformations*
Similarity Lookup (No Transformation)

Jaccard ≥ 2/3

Intersection Size

Union Size

0/10
Similarity Lookup (No Transformation)

\[ r_l \quad A \quad B \quad C \quad D \quad E \quad P \quad q \quad x \quad S \quad t \quad s_l \]
\[ A \quad C \quad D \quad E \quad c \quad s_2 \]
\[ a \quad C \quad E \quad H \quad l \quad s_3 \]

Jaccard ≥ 2/3

Intersection Size

Union Size
Similarity Lookup (No Transformation)

\[
\begin{align*}
rl & \quad A & B & C & D & E & \quad P & q & x & S & t & s1 \\
& A & C & D & E & c & \quad s2 \\
a & C & E & H & I & \quad s3 \\
\begin{array}{ccccccc}
A & a & C & c & D & E & t \\
s2 & s3 & s2 & s2 & s2 & s2 & s1 \\
s3 & & s2 & & s3 & & \end{array}
\end{align*}
\]
Similarity Lookup (No Transformation)
Similarity Lookup (No Transformation)
Similarity Join (No Transformation)
Running Example

Tokens/Words: A, B, C, ..., Z, a, b, ..., z

Transformation Rules:

A → a
B → b
C → c
... ... ...
Z → z
Similarity Lookup

\[
\begin{array}{cccccc}
    r11 & A & B & C & D & E \\
r12 & a & B & C & D & E \\
r13 & A & b & C & D & E \\
\ldots \\
    P & q & x & S & t \\
p & q & x & S & t \\
P & q & x & s & t \\
\ldots \\
    A & C & D & E & c \\
a & C & D & E & c \\
A & c & D & E & c \\
\ldots \\
    a & C & E & H & l \\
a & c & E & H & l \\
a & C & e & H & l \\
\ldots \\
\end{array}
\]
Inverted Index (Naïve)

Cost of matching \( r/l \): \( 32 \times 5 = 160 \) index lookups
Inverted Index (Compressed)
Inverted Index (Compressed)
Inverted Index (Compressed)

\[ r l \quad A \quad B \quad C \quad D \quad E \]

\[
\begin{array}{llllll}
A & a & C & c & D & E \\
s2 & s2 & s2 & s2 & s2 & s2 \\
s3 & s3 & s3 & s3 & s3 & s3 \\
\end{array}
\]
Inverted Index (Compressed)

Cost of matching $r_1$: 10 index lookups + 2 similarity computations

{A, a, B, b, C, c, D, d, E, e}
Token Clustering

A
s2
s3

a
s2
s3

C
s2
s3

c
s2
s3

D
s2

E
s2
s3

e
s2

A

C
D

c
d

E

e
Token Clustering
Token Clustering

Diagram:

- Red circle: A
- Blue circle: E
- Brown circle: D
- Blue triangle: C
- Blue line: s2
- Blue line: s3
- Black line: a

Legend:
- C: s2
- c: s2
- D: s2
- E: s2
- e: s2

Token Clustering
Token Clustering

Cost of matching $rI$: 3 index lookups + 2 similarity computations

$A \quad B \quad C \quad D \quad E$

{A, a, B, b, C, c, D, d, E, e}

![Diagram of token clustering with indices and costs](image-url)
Representative Performance

- **Bing Maps data:**
  - 10M addresses
  - > 24M transformations (mostly programmatic – edit, abbreviations)
  - Average lookup time ~3ms
Conclusion

- Programmable similarity for record matching

- Advantages:
  - Customizability
  - Single similarity function
    - Software engineering advantages
  - Efficient Indexing
References & Acknowledgments

- Arvind Arasu, Venkatesh Ganti, Raghav Kaushik: Efficient Exact Set-Similarity Joins. VLDB 2006: 918-929

Acknowledgments:

Surajit Chaudhuri, Zhimin Chen, Kris Ganjam, Venky Ganti, Raghav Kaushik, Vivek Narasayya, Dong Xin