An Efficient Sliding Window Approach for Approximate Entity Extraction with Synonyms

Jin Wang (UCLA)
Chunbin Lin (Amazon AWS)
Mingda Li (UCLA)
Carlo Zaniolo (UCLA)
OUTLINE

- Motivation
- Preliminaries
- Framework and Techniques
- Experiments
- Conclusion
Dictionary of Entities

Isaac Newton
- English
- physicist
- mathematician
- astronomer
- natural philosopher
- alchemist
- theologian

Sigmund Freud
- Austrian
- psychiatrist
- sociologist

Documents

1 Sir IsaacNewton was an English physicist, mathematician, astronomer, natural philosopher, alchemist, and theologian and one of the most influential men in human history. His Philosophiæ Naturalis Principia Mathematica, published in 1687, is by itself considered to be among the most influential books in the history of science, laying the groundwork for most of classical mechanics.

2 Sigmund Freud was an Austrian psychiatrist who founded the psychoanalytic school of psychology. Freud is best known for his theories of the unconscious mind and the defense mechanism of repression and for creating the clinical practice of psychoanalysis for curing psychopathology through dialogue between a patient and a psychoanalyst.
**Example Application: product search**

<table>
<thead>
<tr>
<th>Dictionary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canon PowerShot G7 X digital camera</td>
</tr>
<tr>
<td>Acer Swift 3 laptop</td>
</tr>
<tr>
<td>......</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Canon G7 X offers a superb image processing......</td>
</tr>
<tr>
<td>......</td>
</tr>
<tr>
<td>PowerShot G7 X captures stunning HD video......</td>
</tr>
</tbody>
</table>
LIMITATIONS OF AEE

- Strings with low syntactic similarity can still be similar!

<table>
<thead>
<tr>
<th>Dictionary</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
</tr>
<tr>
<td>e2</td>
</tr>
<tr>
<td>e3</td>
</tr>
<tr>
<td>e4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>...... When first observed the patient was in shock and had signs of cerebral malaria(^1), disseminated intravascular coagulation, (^2) and acute respiratory distress syndrome, (^3) which in the following 2 days were complicated by acute renal failure (^4)</td>
</tr>
</tbody>
</table>
SYNONYM RULES

• Goal
  – Improve the quality of AEE
  – Combine the semantics carried by synonyms with the syntactic similarity

• Examples
  – Abbreviation
    University of California, Los Angeles ↔ UCLA
  – Same identity
    disseminated intravascular coagulation ↔ consumption coagulopathy
# Approximate Entity Extraction with Synonyms

- **Example:** Institute Name in DB World

<table>
<thead>
<tr>
<th>Dictionary</th>
<th>Synonym rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>$e_1$ Google USA</td>
<td>$r_1$ AU $\Leftrightarrow$ Australia</td>
</tr>
<tr>
<td>$e_2$ University of Chicago USA</td>
<td>$r_2$ Univ. $\Leftrightarrow$ University</td>
</tr>
<tr>
<td>$e_3$ UQ AU</td>
<td>$r_3$ UQ $\Leftrightarrow$ University of Queensland</td>
</tr>
<tr>
<td>$e_4$ UW USA</td>
<td>$r_4$ UW $\Leftrightarrow$ University of Washington</td>
</tr>
</tbody>
</table>

## Document (VLDB 2018 Research Track PC members)

- Dan Ports (Univ. of Washington USA), Haryadi Gunawi (Univ. of Chicago USA), Sandeep Tata (Google USA), Xiaofang Zhou (University of Queensland Australia)

......
OUTLINE

• Motivation
• Preliminaries
• Framework and Techniques
• Experiments
• Conclusion
SET-BASED SIMILARITY

• Common similarity functions:

  - Jaccard: \[ J(x, y) = \frac{|x \cap y|}{|x \cup y|} \geq t \]

  - Cosine: \[ C(x, y) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \cdot \|\vec{y}\|} \geq t \]

  - Dice: \[ D(x, y) = \frac{2|x \cap y|}{|x| + |y|} \geq t \]

<table>
<thead>
<tr>
<th>x = {A,B,C,D,E}</th>
<th>4/6 = 0.67</th>
</tr>
</thead>
<tbody>
<tr>
<td>y = {B,C,D,E,F}</td>
<td>4/5 = 0.8</td>
</tr>
<tr>
<td></td>
<td>8/10 = 0.8</td>
</tr>
</tbody>
</table>
Basic Terminology

- **Entity**: UW USA
  - 1. UW<-> University of Washington
  - 2. UW <-> University of Waterloo
  - 3. USA <-> United States of America

- **Applicable rule set** \( \mathcal{A}(e) \) \{ \{1,3\}, \{2,3\} \}

- **Derived Entity** \( e^i \)
  - The combination of rule applications
    - In above example: UW United States of America
  - Given an entity e, its set of derived entities \( \mathcal{D}(e) \)

- **Derived Dictionary**
  - Given the original dictionary \( \mathcal{E}_0 \)
  
  \[
  \mathcal{E} = \bigcup_{e \in \mathcal{E}_0} \mathcal{D}(e)
  \]
**Problem Formulation**

- **Similarity metrics:** Given an entity \( e \) and a substring \( s \), Asymmetric Rule-based Jaccard is defined as:

\[
J\text{ACCAR}(e, s) = \max_{e^i \in \mathcal{D}(e)} JAC(e^i, s)
\]

- **Approximate Entity Extraction with Synonyms:** Given a dictionary of entities \( E \), a set of synonym rules \( R \), a document \( d \) and the similarity threshold \( \tau \), the goal is to return all the \((e, s)\) pairs where \( s \) is a substring of \( d \) and \( e \in E \) s.t. their JaccAR similarity is no smaller than \( \tau \)
OUTLINE

• Motivation
• Preliminaries
• Framework and Techniques
• Experiments
• Conclusion
**Overall Framework**

**Offline index building**

- Dictionary
- Synonyms

**Online approximate entity extraction**

- Filter
- Verifier

- Document

- Inverted Indexes

- candidates

- results
**PREFIX FILTER [CHAUDHURI ET AL. 2006]**

- Sort the tokens by a **global ordering**
  - E.g. increasing order of document frequency
- Only need to index the first few tokens (prefix) for each record
- Example:
  - jaccard $t = 0.8 \rightarrow |x \cap y| \geq 4$ if $|x| = |y| = 5$
  - $x =$
  - $y =$
    - $y =$
    - $x =$
      - Must share at least one token in prefix to be a candidate pair
        - For jaccard, **prefix length** = $|x| \times (1 - t) + 1 \rightarrow$ each $t$ is associated with a prefix length
INDEX STRUCTURE

• Support prefix filter and **length filter**
  – If the length difference between two strings are beyond a range, they cannot be similar

• Group by length and original entity

<table>
<thead>
<tr>
<th>token $t_1$</th>
<th>......</th>
<th>token $t_a$</th>
<th>......</th>
<th>token $t_n$</th>
</tr>
</thead>
</table>

derived-entity $id_v$

deposit $pos_v$ of token $t_a$ in derived entity with $id_v$

derived entities from $e_m$

length $i$ group

length $j$ group

derived entities from $e_n$
**INDEX STRUCTURE: EXAMPLE**

(a) Global token ordering (based on frequency of occurrences)

Google (1) < Univ. (1) < UW (1) < Washington (1) < Waterloo (1) < AU (2) < Australia (2) < Chicago (2) < Queensland (2) < UQ (2) < University (5) < of (6) < USA (6)

(b) Ordered derived entities

- $e_1^1$: Google USA
- $e_1^2$: Chicago University of USA
- $e_2^2$: Univ. Chicago of USA
- $e_2^3$: AU UQ
- $e_3^3$: Australia UQ
- $e_3^4$: AU Queensland University of
- $e_4^3$: Australia Queensland University of
- $e_4^4$: UW USA
- $e_4^5$: Washington University of USA
- $e_4^6$: Waterloo University of USA

(d) Inverted index

<table>
<thead>
<tr>
<th>Google</th>
<th>……</th>
<th>University</th>
<th>……</th>
<th>USA</th>
</tr>
</thead>
</table>

- derived-entity id
- position of token “university” in derived entity $e_2^1$
- length 4 group
- entities derived from $e_2$
- entities derived from $e_3$
- entities derived from $e_4$
CANDIDATE GENERATION

• Terminology

Naïve Approach
  – Enumerate Substrings and apply prefix filter
  – Bound the window size with length filter

Improving pruning power
  – Dynamic Prefix Computation
    – Window Extend
    – Window Migrate
  – Lazy Candidate Generation
    – Core idea: Scan the inverted list for each token only once
**Dynamic Prefix Computation**

- Window Extend

(a) Extend window from $\mathcal{W}_p^l$ to $\mathcal{W}_p^{l+1}$

(b) Extend window from $\mathcal{W}_3^3$ to $\mathcal{W}_3^4$

(b) Extend window from $\mathcal{W}_3^4$ to $\mathcal{W}_3^5$
**Dynamic Prefix Computation**

- **Window Migrate**

\[ t_{old} = d[p] \]
\[ t_{new} = d[p + l + 1] \]

(b) Migrate window from \( W^l_p \) to \( W^l_{p+1} \)

(c) Migrate window from \( W^4_3 \) to \( W^4_4 \)
OUTLINE

- Motivation
- Preliminaries
- Framework and Techniques
- Experiments
- Conclusion
EXPERIMENT SETUP

• Real world datasets

|        | # docs | # entities | # synonyms | avg $|d|$ | avg $|e|$ | avg $|\mathcal{A}(e)|$ |
|--------|--------|------------|------------|------|------|-------------|
| PubMed | 8,091  | 370,836    | 24,732     | 187.81| 3.04 | 2.42        |
| DBWorld| 1,414  | 113,288    | 1,076      | 795.89| 2.04 | 3.24        |
| USJob  | 22,000 | 1,000,000  | 24,305     | 322.51| 6.92 | 22.7        |

• Environment
  – C++, GCC 4.8.4.
  – 16GB RAM, Ubuntu 14.04

• Evaluation metrics
  – Effectiveness: **Precision**, **Recall**, **F1 score**
  – Efficiency: Query Time
EFFECTIVENESS

• Baseline methods
  – Jaccard
  – Fuzzy Jaccard (FJ) [Wang et al. 2011]: considering edit similarity

• Sample Ground Truth

<table>
<thead>
<tr>
<th>PubMed</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Entity:</strong> moschkowitz disease</td>
</tr>
<tr>
<td><strong>Synonyms:</strong></td>
</tr>
<tr>
<td>moschkowitz disease ⇔ familial thrombotic thrombocytopenia purpura</td>
</tr>
<tr>
<td>moschkowitz disease ⇔ thrombotic thrombocytopenic purpura</td>
</tr>
<tr>
<td><strong>Document:</strong> “... the diagnostic challenge of acquired thrombotic thrombocytopenic purpura in children ...”</td>
</tr>
</tbody>
</table>

| Jaccard = 0.0 | FJ = 0.0 | JaccAR = 1.0 |
EFFECTIVENESS

- Results

<table>
<thead>
<tr>
<th>$\theta$</th>
<th>Jaccard</th>
<th>PubMed</th>
<th>FJ</th>
<th>JaccAR</th>
<th>PubMed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>0.7</td>
<td>0.24</td>
<td>0.73</td>
<td>0.36</td>
<td>0.35</td>
<td>0.73</td>
</tr>
<tr>
<td>0.8</td>
<td>0.14</td>
<td>0.88</td>
<td>0.24</td>
<td>0.34</td>
<td>0.77</td>
</tr>
<tr>
<td>0.9</td>
<td>0.12</td>
<td>0.92</td>
<td>0.21</td>
<td>0.28</td>
<td>0.85</td>
</tr>
</tbody>
</table>

Our method has the best performance since it can capture the semantics contained in synonym rules.
EFFICIENCY: END-TO-END RESULT

- Extending state-of-the-art methods
  - FaerieR [Deng et al. 2015]

Our method outperforms the best existing method by one to two orders of magnitude
EFFICIENCY: FILTERING METHODS

Average Query Time

Number of Accessed Items
EFFICIENCY: SCALABILITY

for $\tau=0.75$, our method took 43.26 ms for 200k entities  
62.71 ms for 600k entities  
125.52 ms for 1m entities
OUTLINE

• Motivation
• Preliminaries
• Framework and Techniques
• Experiments
• Conclusion
CONCLUSION

• A new problem: AEES
• A filter-and-verification framework
  – Clustered indexing structures
  – Effective pruning techniques
• Experimental results show that our methods significantly outperform existing methods
Thank you!

Q & A