Web Based Linkage

Ergin Elmacioglu[1] ergin@psu.edu
Min-Yen Kan[2] kanmy@comp.nus.edu
Dongwon Lee[1] dongwon@psu.edu
Yi Zhang[3] yiz@soe.ucsc.edu

[1]The Pennsylvania State University, USA
[2]National University of Singapore, Singapore
[3]University of California, Santa Cruz, USA

Motivation

- Real world examples of the linkage problem

ACM Digital Library

IMDB & Wikipedia

Name Linkage

- Problem Definition:
  - The process of detecting and merging duplicate *named entities* that represent the same real-world *object*
- Other real world examples
  - Electronic devices (Apple iPod Nano 4GB vs. 4GB iPod nano)
  - Automobile models (Honda Fix vs. Honda Jazz)
  - Companies (T-Fal vs. Tefal)
  - Person names
    - Customs (Jane Doe vs. Doe, Jane)
    - Marriage (Carol Dusseau vs. Carol Arpaci-Dusseau)
    - Misc. (Sean Engelson vs. Shlomo Argamon)

Name Linkage using Collective Knowledge

- There are many effective solutions if enough and discriminative contents are available
  - E.g., Contents based record linkage techniques
- Becomes challenging when
  - Not enough content to compare
  - Content does not help to identify

- Proposal: use external knowledge
  - ***Ask people what they think***
  - Collective knowledge of people from the Web

Name Linkage using Collective Knowledge

- Main Idea: Use the Web as a collective knowledge of people in solving the Name Linkage Problem

- Hypothesis:

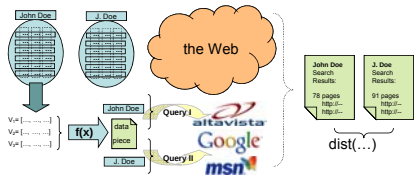
If an entity **e1** is a duplicate of another entity **e2**, and if **e1** frequently appears together with information **I** on the Web, then **e2** may appear frequently with **I** on the Web, too.

Small Test

- Search results from Google:

"Jeffrey D. Ullman"	384,000 pages	
"Jeffrey D. Ullman" + "aho"	174,000 pages	45%
"J. Ullman"	124,000 pages	
"J. Ullman" + "aho"	41,000 pages	33%
"Shimon Ullman"	27,300 pages	
"Shimon Ullman" + "aho"	66 pages	0%

Web Based Linkage: Overview



Step 1. Select Representative Data

- What to select
  - A single token "aho"
  - A key phrase "stanford professor"
  - A sentence or more?
- How to select
  - Assess importance
    - tf, tf*idf, latent topic models, …
- How many to select
  - 1, 2, … n
- Where to select from?
  - Contents of canonical entity, variant, both

Step 2. Acquire Knowledge from Web

- How to form the query?
  - Single information "I" (the most important data piece)
    - `"Jeffrey D. Ullman" AND "Aho"`
  - Multiple information "I_1", "I_2", "I_3", … (the most ***k*** important data pieces)
    - Conjuction:
      `"Jeffrey D. Ullman" AND "Aho" AND "database" AND "vldb"…`
    - Disjuction:
      `"Jeffrey D. Ullman" AND ("Aho" OR "database" OR "vldb"…`
    - Hybrid:
      `"Jeffrey D. Ullman" AND "Aho" AND ("database" OR "vldb"…`

Step 3. Interpret the Collective Knowledge

For entities e_c, e_i and information t_c

- Page Cou

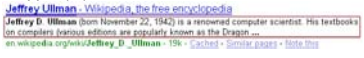
$sim(e_c,$	Jeffrey D. Ullman J. Ullman portal.acm.org infolab.stanford.edu en.wikipedia.org theory.lcs.mit.edu = 4/16	Jeffrey D. Ullman Shimon. Ullman portal.acm.org = 1/19
$sim(e_c,$		$\cap t_c)$

- URLs

$sim(e_c,$	Jeffrey D. Ullman J. Ullman =1/(174,000 - 41,000)	Jeffrey D. Ullman Shimon Ullman =1/(174,000 - 66)
		$\cap t_c)\|$

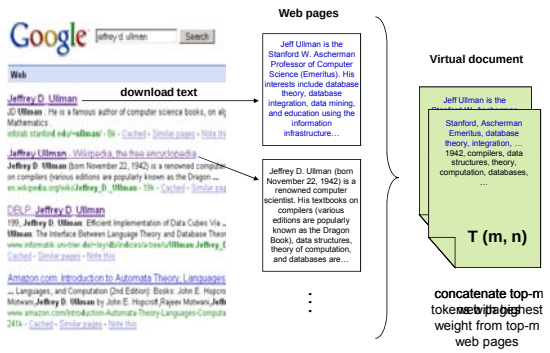
Step 3. Interpret the Collective Knowledge

- Web Page Contents
  - Use top-k returned Web pages for each entity
  - Represent each set by a **Virtual Document**
  - Some heuristics
    - D (m): Top m (≤ k) documents are concatenated
    - T (all, n): Top n tokens with the highest weight from all top-k web pages
    - Snippet (m): Snippets of top m (≤ k) web pages



    - Probabilistic Language Model: KL-divergence
  - $sim(e_c, e_i)$ = doc_sim($vdoc(e_c)$, $vdoc(e_i)$)

Virtual Document Creation



Experimental Validation

Presented Next

- Tested against
  - ACM , ArXiv, IMDB
- Variations tested
  - Step 1: single token, top-k tokens
  - Step 2: conjunctive query only
  - Step 3
    - Page count, URL
    - Virtual Document: 10 heuristics and 2 language models
- Search Engines
  - Google, MS Live Search

Experimental Validation

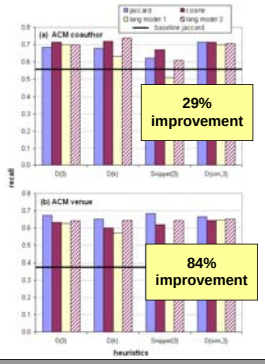
ACM data set:

- **43 authors**
  - 14.2 citations/author
- **21 candidates/block**
  - 3.1 citations/candidate
- **1.8 name variants/block**
  - 6.7 citations/variant

Jeffrey D. Ullman

Foto N. Afrati, Chen Li, Jeffrey D. Ullman Using views to generate efficient evaluation plans for queries. **J. Comput. Syst. Sci. 73: 703-724 (2007)**
.....

foto, n, afrati, chen, li, ...	using, views, generate, efficient, ...	j, comput, syst, sci, 73, ...
coauthor	**title**	**venue**



Experimental Validation

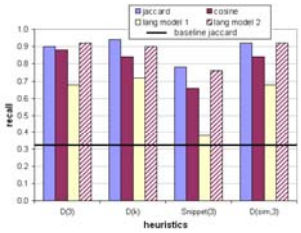
IMDB data set:

- **50 actors**
  - 24 titles/entity
- **20 candidates/block**
  - 24 titles/candidate
- **1 name variant/block (a.k.a. name)**
  - 23.5 titles/variant

Christina Aguilera

Moulin Rouge
What Women Want
The Bold and the Beautiful
The Next Best Thing
.....

moulin, rouge, bold, beautiful, ...	what,women, want, next, best, thing, ...
Christina Aguilera	**Xtina (a.k.a.)**



193% improvement

Scalability

- Not scalable:
  - A large number of Web accesses
  - Network traffic, load of search engine and web sites

Running times of the experiments on the ACM data set using the *tf* scheme on the title attribute

Methods	Recall	Running time	# of Web accesses
Baseline - jaccard on the current content	0.495	0.19 min	0
Googling - cosine using $D(3)$ and Google	0.653	290.44 min	7951
Googling - jaccard using $D(3)$ and Google	0.711	289.32 min	7951
Googling - lang. model 1 using $D(3)$ and Google	0.573	289.29 min	7951
Googling - lang. model 2 using $D(3)$ and Google	0.756	289.47 min	7951
Googling - cosine using $D(3)$ and the local snapshot	0.463	5.53 min	0
Googling - jaccard using $D(3)$ and the local snapshot	0.445	5.38 min	0
Googling - lang. model 1 using $D(3)$ and the local snapshot	0.457	5.37 min	0
Googling - lang. model 2 using $D(3)$ and the local snapshot	0.494	5.39 min	0

- Solutions:
  - A better blocking scheme
  - Local snapshot of the Web
    - Stanford WebBase Project
    - ~100 million web pages from >50,000 sites including many .edu domains
    - Downloaded the half of the data & filtered
    - Local snapshot containing 3.5 million relevant pages

Related Work

- Abundant research on related problems
  - DB: approximate join, merge/purge, record linkage
  - DL: citation matching, author name disambiguation
  - AI: identity uncertainty
  - LIS: name authority control
- In a nutshell, existing approaches often do:
  - For two entities, *e1* and *e2*, capture their information in data structures, *D(e1)* and *D(e2)*
  - Measure the distance or similarity between data structures: *dist(D(e1), D(e2)) = d*
  - Determine for matching:
    - If *d < threshold*, then *e1* and *e2* are matching entities
- Work well for common applications
- Ours performs better when
  - Entities lack useful information

Conclusion & Future Work

- The Name Linkage Problem
  - incomplete, noisy, non-descriptive data
  - usage of the Web to get additional information for the linkage process
- Future Work
  - A formal framework
  - Extension to "name disambiguation"
  - Experimentation on more & (larger) data sets
  - Scalability