Ranking-Based Name Matching for Author Disambiguation in Bibliographic Data

Jialu Liu, Kin Hou Lei, Jeffery Yufei Liu, Chi Wang, Jiawei Han
Presenter: Chi Wang

UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

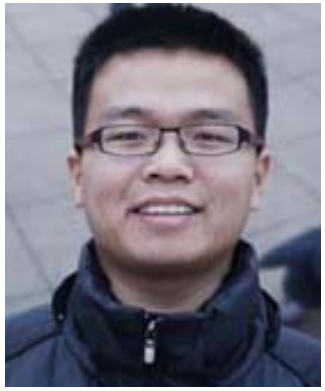
Background

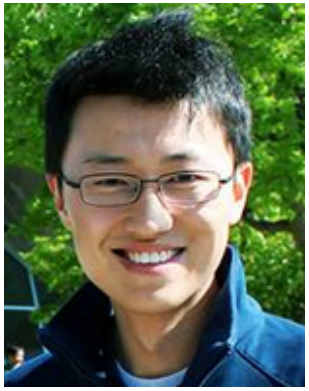
Team name: SmallData

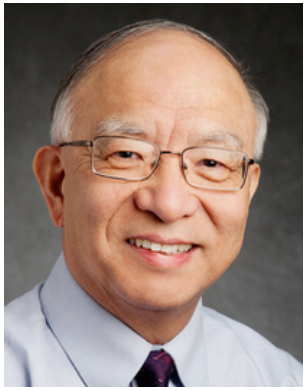
Achievement: 2nd @ 2nd Track

Performance: 99.157 (F1 score)

From: CS & STAT @ UIUC







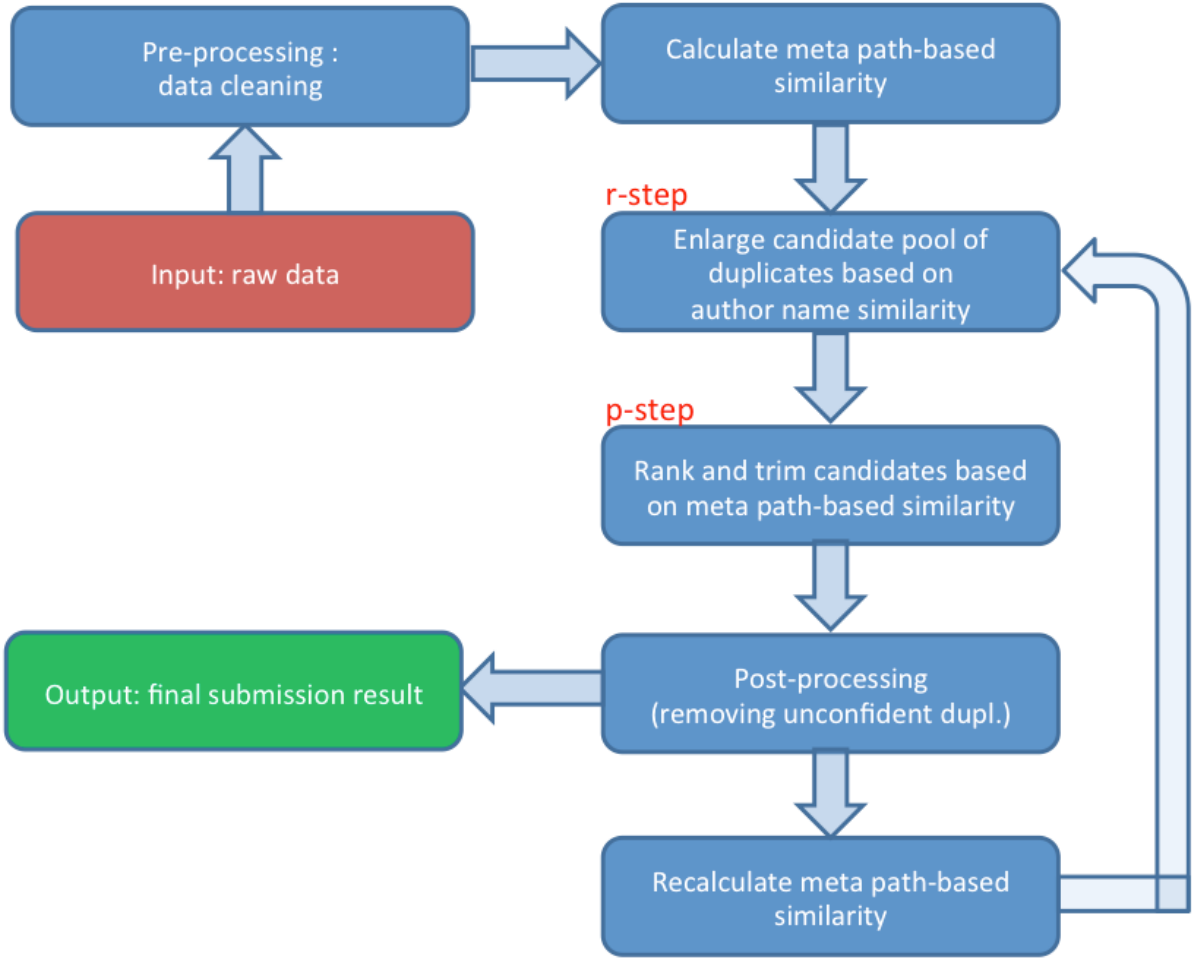
Outline

- Overview
- Details of RankMatch
- Experiment
- Discussion

Challenge

- No training data
- Noises in the data set
  - Spelling, Parser, etc.
- Names from different areas
  - Asian, Western
- Test ground truth not trustable

Overview of the System (RankMatch)



Outline

- Overview
- Details of RankMatch
- Experiment
- Discussion

Pre-process: Data Cleaning

- Noisy First or Last Names
  - *Eytan H. Modiano* and *Eytan Modianoy*
  - *Nosrat O. Mahmoodo* and *Nosrat O. Mahmoodiand*
- Mistakenly Separated or Merged Name Units
  - *Sazaly Abu Bakar* and *Sazaly AbuBakar*
  - *Vahid Tabataba Vakili* and *Vahid Tabatabavakili*
- Way to Recover
  - Build statistics of name units
    - Count[“Modianoy”] << Count[“Modiano”]
    - Count[“Tabataba” & “Vakili”] > Count[“Tabatabavakili”]

The r-Step: Improving Recall

- Improving the recall of the algorithm means that given an author ID (input), one should find as many potential duplicates (output) as possible.
- What do we need to consider?

Name!

- String-based Consideration
  - Levenshtein Edit Distance
    - Levenshtein edit distance between two strings is the minimum number of single character edits required to change one string into the other.
    - Spelling or OCR error
  - Soundex Distance
    - Soundex algorithm is a phonetic algorithm that indexes words by their pronunciation in English.
    - “Michael”,“Mickel” and “Michal”
  - Overlapping Name Units
    - Name reordering brought by parser
    - *Wing Hong Onyx Wai* and *Onyx Wai Wing Hong*

- Name-Specific Consideration
  - Name Suffixes and Prefixes
    - Prefixes: “Mr”, ”Miss”
    - Suffixes: “Jr”, “I”, “II”, “Esq”
  - Nicknames
    - “Bill” and “William”
    - No transitive rule: “Chris” could be a nickname of “Christian” or “Christopher” but “Christian” will not be compatible with “Christopher”.
  - Name Initials
    - In research papers, people always use initials.
    - *Kevin Chen-Chuan Chang* and *K. C.-C. Chang*, *Kevin C. Chang*
    - Together with nicknames, “B” and “W” can be compatible because they can represent “Bill” and “William”

- Name-Specific Consideration (Cont.)
  - Asian Names and Western Names
    - Different areas have totally different name rules.
    - For example, East Asians usually lack the middle names and their first and last names could contain more than one name unit.
      - *Andrew Chi-Chih Yao* and *Michael I. Jordan*
    - So the thresholds for two name strings to be viewed as similar in terms are different for different areas.
    - For example, for edit distance
      - *Mike Leqwis* and *Mike Lewis*
      - *Wei Wan* and *Wei Wang*
    - Lots need to be done in this direction!

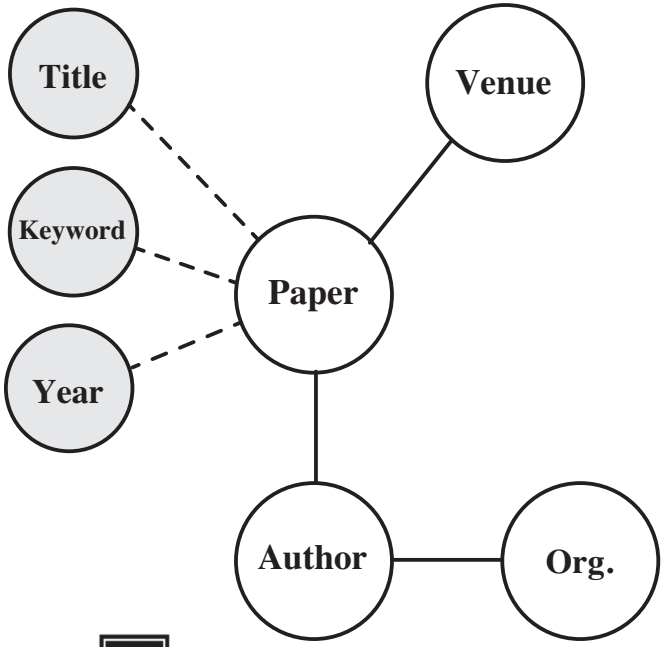
- Efficiency Consideration
  - To find potential duplicate author ID pairs, the ideal way is to process any pairs of author IDs in the dataset which is of time complexity $O(n^2)$.
    - Doable using MapReduce
  - We choose to reduce the search space via mapping author names into pools of name initials and units so that we only compare the pairs within the same pools.
    - *Michael Lewis* -> Pool[“Michael”], Pool[“Lewis”], Pool[“ML”]
    - Lossy!
    - Transitive rule: if name string a is similar to b and b is similar to c, then the name pair a and c needs to be checked to see whether they are similar or not.

The p-Step: Improving Precision

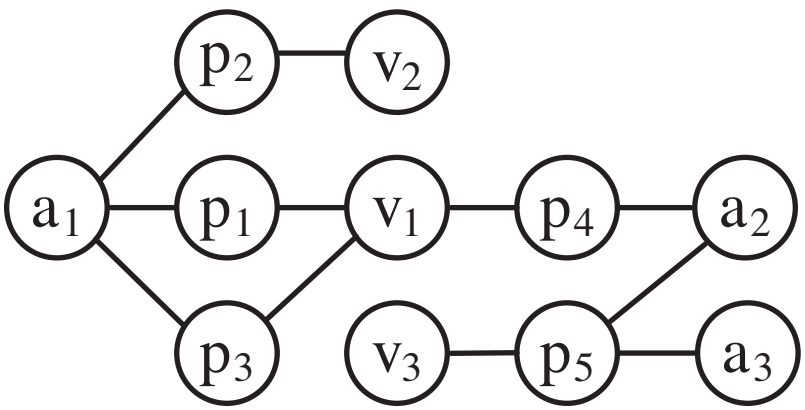
- Improving the precision of the algorithm means that once finding potential duplicates (input) from r-step, we need to infer the real author entity (output) shared by one or more author IDs.
- What do we need to consider?

Network!

- Meta-path in networks
  - A meta-path P is a path defined on the graph of a network schema. For example, in this competition data set, the co-author relation can be described using the length-2 meta-path APA (author-paper-author)



- Adjacency Matrix for sub-networks
  - Adjacency matrix is a means of representing which nodes of a network are adjacent to which other nodes. Here is an example of adjacency matrices for Author-Paper and Paper-Venue separately.



$$M_{A,P} \quad \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{ccccc} p_1 & p_2 & p_3 & p_4 & p_5 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$M_{P,V} \quad \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

- Measure Matrix for Nodes Similarity
  - A measuare matrix is for keeping similarities for any pair of nodes based on a meta-path.
  - For example, the measure matrix for Author-Paper-Venue is:

$$\overline{M_{A,V}} = Normalize(M_{A,P} \times M_{P,V})$$

    - L_2 Normalization is applied to make such that the self-maximum property can be achieved.
  - Similarly, the measure matrix for APVPA is:

$$\overline{M_{A,A}} = \overline{M_{A,V}} \times \overline{M_{A,V}}^T = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{ccc} a_1 & a_2 & a_3 \end{array} \\ \begin{bmatrix} 1.0000 & 0.6325 & 0 \\ 0.6325 & 1.0000 & 0.7071 \\ 0 & 0.7071 & 1.0000 \end{bmatrix} \end{array}$$

- Multiple Measure Matrices
  - We are interested in similarity score between authors
  - Such score can be obtained via multiple measure matrices with different meta-paths.
  - To support measure matrices defined on different meta-paths, we adopt the linear combination strategy:

$$Sim(a_i, a_j) = \sum W_{path} Sim_{path}(a_i, a_j)$$

    - The selected meta-paths are APA, AOA, APAPA, APV PA, APKPA, APTPA and APY PA. The weights for them are decreasing progressively.

- Ranking-based Merging
  - Assume we have three authors and their similarity scores in the listed tables
  - To infer the real entity behind each ID
    - Sort the similarity scores
    - Start merging from top ranked ID
      - (2), (3) are in conflict, skip
      - (1), (2) merge -> (1, 2)
      - (1), (3) are in conflict because (2) and (3)
      - return (1, 2) and (3)
    - Once two IDs have multiple publications and low meta-path-based similarity score, reject their merging request.

Name	Author ID	Matched IDs
Michael Lewis	1	2, 3
Michael J. Lewis	2	1
Michael P. Lewis	3	1

Author ID Pair	Similarity	Rank
$(1, 2)$	0.6325	2
$(1, 3)$	0	3
$(2, 3)$	0.7071	1

- Ranking-based Merging (cont.)
  - Expand author names corresponding to the IDs once we are confident about two IDs to be the duplicate.
    - For example, as authors 1 and 2 are highly possible to be the same person and the name of author 2 has better quality than that of author 1, we can replace the name of author 1 to be *Michael J. Lewis*.
    - Suppose the full name of author 1 or 2 to be *Michael James Lewis* and we have a new author with name *James Lewis*.
    - If we do not adopt this name expanding mechanism, obviously author 1 and this new author are in conflict.

Name	Author ID	Matched IDs
Michael Lewis	1	2, 3
Michael J. Lewis	2	1
Michael P. Lewis	3	1

Author ID Pair	Similarity	Rank
$(1, 2)$	0.6325	2
$(1, 3)$	0	3
$(2, 3)$	0.7071	1

Post-processing

- "Unconfident" duplicate author IDs should be removed even though their names are compatible and their meta-path-based similarity scores are acceptable.
- We define "unconfident" to have two factors
  - the difference between name strings in terms of unmatched name units to be large
  - the meta-path-based similarity score to be not large.
  - *Wing Hong Onyx Wai* and *W. Hong*

Iterative Framework

- The iterative framework takes the detected duplicates of the last iteration as part of the input.
- There are two reasons to do this:
  - It help generate better meta-path-based similarity scores by merging “confident” duplicate author IDs.
  - With the name expansion in p-step, the original input has changed and we need to rerun the algorithm.
- Time consuming

Outline

- Overview
- Details of RankMatch
- Experiment
- Discussion

Basic Information

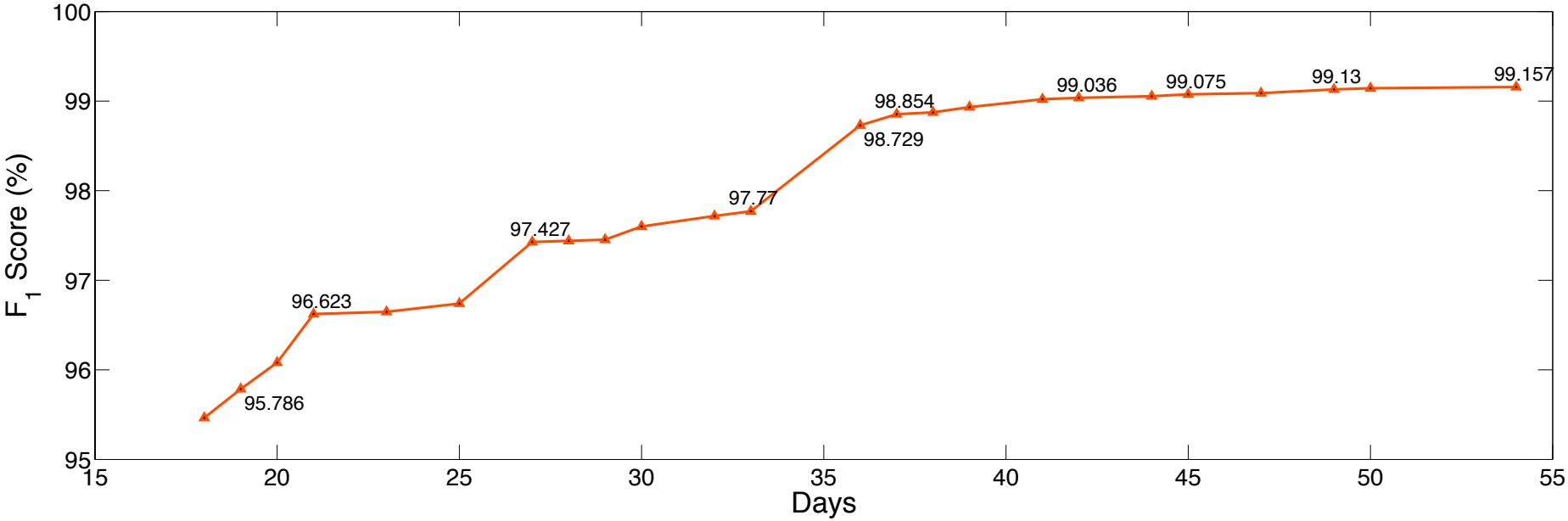
- Environment: PC with Intel I7 2600 and 16GB memory
- Language: Python 2.7
- Time Consumption: One hour for one iteration
- Code: https://github.com/remenberl/KDDCup2013

Name Compatible Test

Author Name A	Author Name B	Expected Result
Jiawei Han	*Jia Han*	In Conflict
Xiang Li	*Xiang Lin*	In Conflict
Gordon D. Moskowitz	*Gordon Blaine Moskowitz*	In Conflict
H. Murray-Rust	*D. M. Murray-Rustt*	In Conflict
Deliang L. Wang	*Liang Wang*	In Conflict
Takeshi Mori	*Taketoshi Mori*	In Conflict
Tadashi Suzuki	*Takashi Suzuki*	In Conflict
Hong-Hu Zhu	*H. H. Zhu*	Compatible
Ralph Mac Nally	*RalphMac Nally*	Compatible
V. Scott Gordont	*V. Scott Gordon*	Compatible
Jeff W. Hughes	*Jeffrey W. Hughes*	Compatible
William Hughes	*Bill Hughes*	Compatible
William Hughes	*B. Hughes*	Compatible
Valli Kumari Vatsavayi	*V. Valli Kumari*	Compatible
Mercedes Femandez-Redondo	*Mercedes Fernandez Redondo*	Compatible
Aliaa Abdel-Haleim Abdel-Razik Youssif	*Aliaa A. A. Youssif*	Compatible

Improvement of Performance

- Met bottleneck in the last few days.



Contributions of Modules

- Not accurate

Performance	Gain	New Module(s)	Days
95.376	-	Same Author Name Benchmark	-
95.786	0.410	+ Meta-path: Coauthor	19
96.623	0.837	+ Name Initials, Omitted Middle Name	21
97.427	0.804	+ Meta-path: Covenue	27
97.770	0.343	+ Nicknames + Asian names handling	33
98.729	0.959	+ Accepting name-compatible author pair even with zero meta-path-based similarity	36
99.020	0.291	+ Name reordering + noisy last/first name pre-processing	37
99.036	0.016	+ Rough post-processing	42
99.075	0.039	+ SoundEx distance	45
99.130	0.055	+ Name units breaking/merging pre-process, + name expansion	49
99.157	0.027	+ Iterative framework + more aggressive post-processing	54

Outline

- Overview
- Details of RankMatch
- Experiment
- Discussion

Data

- Lack of training data makes it difficult to evaluate the model, especially the p-step (meta-paths)
- Not able to find an effective way to make use of the training set which is released for Track I
- How are the evaluation set generated: labeled by algorithm or by domain experts?

Promising Directions

- Apply machine learning techniques to train a classifier using features like edit distance, similarity score from measure matrices (needs labels)
- Build models for names from different areas.
  - Indian, Japanese, Arabic and some western languages like French, German, Russian and so on

Conclusion

- String-based name matching to increase recall
- Network-based similarity score to increase precision
- A good chance to combine research insights and engineering implementation

Thanks. Q&A