Results on Tracks 1 and 2 of KDD Cup 2013

Chih-Jen Lin
Department of Computer Science
National Taiwan University

Joint work with members of the team “Algorithm” from National Taiwan University
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
At National Taiwan University, we organized a course for KDD Cup 2013

Three instructors, three TAs, and 18 students

18 students split to six sub-teams named by algorithms
A*, Binary-Search, Dijkstra, K-means, Quick-Sort, Simplex

Submission quotas are equally divided to six sub-teams
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Paper-author Identification

- Given an (author, paper) pair, did the author write the paper?
- What information do we have?
  - Author and paper profiles
  - Labeled (author, paper) pairs
    - Confirmed: author wrote paper
    - Deleted: author didn’t write paper
- Under a given (author, paper), we use target author and target paper to distinguish them from other authors/papers
Submission: ranking query papers for each query author
Example: author 9417 has query papers 1, 2, 3, 6, and 9.
If 3, 6 are confirmed and 1, 2, 9 are deleted, we should submit “9417, 3 6 1 2 9”

MAP (Mean Average Precision) is the evaluation measure
Internal Validation Set

We split `Train.csv` to internal training/validation sets due to the limited number of submissions per day. This also avoids overfitting the leader board.

\[
\begin{array}{ccc}
5 & : & 2 & : & 3 \\
\end{array}
\]

\[
\begin{array}{c}
\text{Train} \\
\text{Valid} \\
\text{Test} \\
\end{array}
\]

\[
\begin{array}{c}
\text{internal} \\
\text{train} \\
\text{valid} \\
\end{array}
\]

randomly shuffled
System Overview

- Feature Generation
  - Individual Models
  - Combining Models
  - Post-Processing

List of **97 features** can be seen in the paper
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Features from Author Profiles

- Given a query \((\text{author}_{1360414}, \text{paper}_{1841516})\). What information do we have about the author?

  - \text{Author.csv}: 1360414, Chih-Jen Lin, National Taiwan University
  
  - \text{PaperAuthor.csv}: 1841516, 1360414, Chih-Jen Lin, "National Taiwan University, Taipei"

- Distance between target author’s names, affiliations, etc. in two csv files \(\Rightarrow\) features to indicate consistency
We need to address two issues
- Distance between full and abbreviated names
- Western and eastern order of names
  Example: “Chih Jen Lin” and “Lin Chih Jen”
- See paper for details
Features from Author Profiles (Cont’d)

- String distance:
  - Jaro
  - Levenshtein
  - Jacard
  - Match ratio

- Features from Author Profiles:
  - Western and eastern orders
  - Full vs. abbreviated
  - Name
  - Affiliation
Features from Coauthors Names

- Example: deleted paper 5633 of Li Zhang has two authors with the same name
- Relation between target author and authors of target paper can be features
- Examples
  1. Minimum name distance between the target author and authors of the target paper
  2. Same as 1, but check abbreviated names
Features for Author/Paper Consistency

- Information should be consistent across papers and authors
- Examples
  1. Maximum distance between target author’s affiliation and affiliations of co-authors in target paper
  2. Maximum distance between target paper’s title and target author’s papers
Missing Value Handling

- Two empty strings have zero distance
  \[ d(\text{‘Chih J Lin’, ’C Jen Lin’) \geq d(”,”) \] 
- Replace distance between empty strings with non-zero value

<table>
<thead>
<tr>
<th>Distance</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jaro</td>
<td>0.5</td>
</tr>
<tr>
<td>Jacard</td>
<td>0.5</td>
</tr>
<tr>
<td>Levenshtein</td>
<td>average length of all entries</td>
</tr>
</tbody>
</table>

- Missing value indicators. Example: number of coauthors without affiliation information
Features Using Publication Time

Examples

1. Earliest/latest publication year of target author
2. Publication year of target paper

Data cleaning:

- Years outside [1800, 2013] are removed
- Then we must handle missing values
Features Based on A Network

- We construct a network of authors, papers, journals, and conferences

Node type:
- a: author
- p: paper
- C: conference
- J: journal
Features Based on A Network (Cont’d)

- From the network we can extract features to describe node relationships

- Examples
  1. # of publications of the author
  2. # of coauthored papers of the target author with all the coauthors of the target paper
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Classification

- Tree-based classifiers
  - Random forests (RF)
  - Gradient boosting decision tree (GBDT)
  - LambdaMART (LM)

<table>
<thead>
<tr>
<th>classifier</th>
<th>tree ensemble</th>
<th># of trees</th>
<th>parallel</th>
<th>MAP on Valid.csv</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF</td>
<td>bagging</td>
<td>12,000</td>
<td>yes</td>
<td>0.983340</td>
</tr>
<tr>
<td>GBDT</td>
<td>boosting</td>
<td>300</td>
<td>no</td>
<td>0.983046</td>
</tr>
<tr>
<td>LM</td>
<td>boosting</td>
<td>300</td>
<td>no</td>
<td>0.983047</td>
</tr>
</tbody>
</table>

RF is sensitive to the initial random seed. Using 12,000 trees stabilizes the results.
Outline

1. Introduction
2. **Track 1: paper-author identification**
   - Feature generation
   - Classification
   - **Ensemble and post-processing**
   - Results
3. **Track 2: author disambiguation**
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Ensemble

- Weighted average over RF, GBDT, and LambdaMart
- Didn’t use more complicated settings like regression because we have only three models
- A simple grid search on weights
- Final weights
  
  \textbf{RF: 5, GBDT: 1, and LambdaMart: 1.}
Our post-processing procedure is simple, but one thing to note is duplicated paper IDs.

If an author has confirmed papers 1,2,2,4 and deleted paper 3

The evaluation code seems to consider the 2nd “2” as a deleted paper

Thus, MAP of 1,2,4,3,2 > MAP of 1,2,2,4,3

We move duplicated ones to the end
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
## Results

<table>
<thead>
<tr>
<th></th>
<th>Public</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st of public</td>
<td>0.98554</td>
<td>0.98100</td>
</tr>
<tr>
<td>12th of public (ours)</td>
<td>0.98235</td>
<td>0.98259 (1st)</td>
</tr>
</tbody>
</table>

- Possible reasons of the best result in the end
  - Improvements after `Valid.csv` is released
    1. Data cleaning: unicode → ASCII
    2. Missing-value handling (0.98334)
    3. Ensemble (0.98390)
  - We didn’t give up even though we were the 12th!
  - We effectively use the internal validation set to avoid overfitting
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Author disambiguation

C. J. Smile Lin
National Taiwan Univ.
LIBSVM Guide

C. J. Cry Lin
Univ. of Michigan
LIBLINEAR Guide

Are they duplicates?
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Main Strategies

- Using **string matching** rather than other learning techniques
- An author without any papers is treated as a single group without duplicates
- Recognizing if an author is **Chinese or not**
Architecture

Implementation 1 → Ensemble → Final
Implementation 2 → Ensemble
## Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Public</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation 1</td>
<td>0.99186</td>
<td>0.99198</td>
</tr>
<tr>
<td>Implementation 2</td>
<td>0.99071</td>
<td>0.99083</td>
</tr>
<tr>
<td>Final</td>
<td>0.99195</td>
<td>0.99202</td>
</tr>
</tbody>
</table>
Framework of the Two Implementations

1. **Cleaning**: remove redundant information
2. **Chinese-or-not**: classify each author as Chinese or non-Chinese
3. **Selection**: select a set of candidates of possible duplicates for each author
4. **Identification**: identify duplicates from the set of candidates for each author
5. **Splitting**: split incorrect cases (not discussed here)
Differences between Two Implementations

- The basic elements are different
  - Implementation 1: author identifiers
  - Implementation 2: author names

<table>
<thead>
<tr>
<th>Author identifier</th>
<th>1001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name in Author.csv</td>
<td>Chih Jen Lin</td>
</tr>
<tr>
<td>Names in PaperAuthor.csv</td>
<td>C. J. Lin, Chih Jen Peter Lin, C. J. P. Lin</td>
</tr>
</tbody>
</table>

- More (complicated) rules in Implementation 1
- Focus on Implementation 1 because of time limitation
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Cleaning

- Clean redundant information.

Examples:

- CHih JEN Lin $\rightarrow$ chih jen lin
- Mr. Chih Jen Lin $\rightarrow$ chih jen lin
- Chih Jén Lin $\rightarrow$ chih jen lin
- Chih Jen Bill Lin $\rightarrow$ chih jen william lin
Chinese and non-Chinese names are very different

- No middle name in Chinese. “Chih Lin” and “Chih J. Lin” are likely different
- Some Chinese last names like “Wang” are too common. Also, “林” and “薺” are romanized to “Lin”
Using common Chinese last names and words as a dictionary
Check if the name contains words in our dictionary

Examples:

- Without full word $\rightarrow$ Non-Chinese; full word: a word without "." and longer than 1 e.g., C J L
- Only one full word and it is in Chinese dictionary $\rightarrow$ Chinese e.g., C. J. Lin
- More than one full word not in Chinese dictionary $\rightarrow$ Non-Chinese e.g., C. J. Chris Peter Lin
Selection

- Find candidates of duplicates to reduce square complexity to linear in future comparison

- Each author generates several keys. “Chih Jen Lin” has:
  - “Chih”
  - “Jen”
  - “Lin”
  - “Chih Jen”
  - “Jen Lin”
  - “Chih Lin”
  - “Chih Jen Lin”
  - “Chih Jen Lin”
Selection – Chih Jen Lin’s candidates

- One is a candidate of another if two share the same key. Ignore common keys.
Selection — Chih Jen Lin’s candidates

Chien Chih Wang
Peng Jen Chen
Shou De Lin
C. J. Lin
Chih Jen Lin
Felix Wu
Yu Chin Juan
Wei Sheng Chin
Yong Zhaung
Lin Chih Jen
Hsiao Yu Tung
Hsuan Tien Lin
Identification

- Find duplicates from candidates
Identification — Matching Functions

13 matching functions

1. Two authors have the same words
2. *(Non-Chinese only)* Only one author has middle name and their last names differ in the last two characters
3. ...

Examples:

<table>
<thead>
<tr>
<th>Two names</th>
<th>Fun.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chih Jen Lin, Lin Chih Jen</td>
<td>1</td>
</tr>
<tr>
<td>Michael I. Jordan, Michael Jordan</td>
<td>2</td>
</tr>
</tbody>
</table>
Making corrections as matching functions may wrongly identify duplicates

Check if two names are “loosely identical”

Examples:

<table>
<thead>
<tr>
<th>Potential duplicates</th>
<th>Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. J. Lin, Chih Jen Lin, Chih Lin, Chen Ju Lin</td>
<td>✗</td>
</tr>
<tr>
<td>C. J. Lin, Chih Jen Lin, Chih J. Lin</td>
<td>✓</td>
</tr>
</tbody>
</table>
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
**Ensemble**

<table>
<thead>
<tr>
<th>Method</th>
<th>Author identifier</th>
<th>Duplicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation 1</td>
<td>10</td>
<td>10, 11</td>
</tr>
<tr>
<td>Implementation 2</td>
<td>10</td>
<td>10, 11, 12, 13, 14</td>
</tr>
<tr>
<td>Ensembled</td>
<td>10</td>
<td>10, 11, 12, 14</td>
</tr>
</tbody>
</table>

- Implementation 1 considered as major predictions
- \{12, 13, 14\} become additional duplicates
- Check if each of (10, 12), (10, 13), (10, 14) has similar affiliations or fields
1. Introduction

2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results

3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis

4. Conclusions
We conduct some analyses after the competition. Thank Kaggle for re-opening the submission site.

<table>
<thead>
<tr>
<th>Method</th>
<th>Public</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td>Final</td>
<td>0.99195</td>
<td>0.99202</td>
</tr>
<tr>
<td>Implementation 1</td>
<td>0.99186</td>
<td>0.99198</td>
</tr>
<tr>
<td>Without Chinese-or-not</td>
<td>0.99109</td>
<td>0.99125</td>
</tr>
<tr>
<td>Without dry-run</td>
<td>0.99097</td>
<td>0.99112</td>
</tr>
<tr>
<td>Without both</td>
<td>0.98891</td>
<td>0.98934</td>
</tr>
</tbody>
</table>

Splitting Chinese/non-Chinese and the dry-run function in the identification stage are useful.
Outline

1. Introduction
2. Track 1: paper-author identification
   - Feature generation
   - Classification
   - Ensemble and post-processing
   - Results
3. Track 2: author disambiguation
   - Strategies and architecture
   - Implementation
   - Ensemble and typo handling
   - Analysis
4. Conclusions
Conclusions

- Our code is available at
  - github.com/kdd-cup-2013-ntu/track1
  - github.com/kdd-cup-2013-ntu/track2
- We thank the organizers and the support from National Taiwan University