NAME MATCHING WITH PHYLOGENIES

Nicholas Andrews, Jason Eisner, Mark Dredze
Martin Freeman
Martin Freeman
Marty Freeman
Martin Freedman
Marty Freemen
Marty Freeman
Martin F
STRING COMPARISON

- Levenshtein distance
  - Edit distance between two strings
- Jaro Winkler
  - Measures matching characters and transpositions
STRING COMPARISON

• Levenshtein distance
  • Edit distance between two strings
• Jaro Winkler
  • Measures matching characters and transpositions

Mark Dredze vs. Mark Drezde (e.g. typo, name variant)

Mark Dredze vs. Benjamin Van Durme
NAME VARIATION

- Nicknames: Benjamin Van Durme vs. Ben Van Durme
- Aliases: Caryn Elaine Johnson vs. Whoopi Goldberg
- Chinese Names: Zhang Wei vs. Wei Zhang
- Arab Names:
  Muhammad ibn Saeed ibn Abd al-Aziz al-Filasteeni vs. Muhammad vs. Abu Kareem
OUR GOAL

LEARN HOW TO MATCH NAMES
Probabilistic finite state transducers encode a probability distribution over strings given a string.

Character operations: copy, substitute, delete, insert.

Train parameters on name pairs.
<table>
<thead>
<tr>
<th>William Ronald Dodds Fairbairn</th>
<th>Ronald Fairbairn</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Ideal: matched name pairs</td>
<td></td>
</tr>
<tr>
<td>William Ronald Dodds Fairbairn</td>
<td>Ronald Fairbairn</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>• Ideal: matched name pairs</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ronald Fairbairn</th>
<th>W. R. D. Fairbairn</th>
</tr>
</thead>
<tbody>
<tr>
<td>William Ronald Dodds Fairbairn</td>
<td></td>
</tr>
</tbody>
</table>

• Sets of matching names
William Ronald Dodds Fairbairn

Ronald Fairbairn

W. R. D. Fairbairn
William Ronald Dodds Fairbairn

Ronald Fairbairn

W. R. D. Fairbairn
William Ronald Dodds Fairbairn

Ronald Fairbairn

W. R. D. Fairbairn
Ronald Fairbairn

William Ronald Dodds Fairbairn

Ronald Fairbairn

W. R. D. Fairbairn
<table>
<thead>
<tr>
<th>William Ronald Dodds Fairbairn</th>
<th>Ronald Fairbairn</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Ideal: matched name pairs</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ronald Fairbairn</th>
<th>W. R. D. Fairbairn</th>
</tr>
</thead>
<tbody>
<tr>
<td>William Ronald Dodds Fairbairn</td>
<td></td>
</tr>
</tbody>
</table>

• Sets of matching names
<table>
<thead>
<tr>
<th>First Name</th>
<th>Last Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ronald</td>
<td>Fairbairn</td>
</tr>
<tr>
<td>Ronald</td>
<td>W. R. D. Fairbairn</td>
</tr>
<tr>
<td>William Ronald Dodds Fairbairn</td>
<td>William Ronald Dodds Fairbairn</td>
</tr>
</tbody>
</table>

- **Sets of matching names**

<table>
<thead>
<tr>
<th>First Name</th>
<th>Last Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Wilkins</td>
</tr>
<tr>
<td>Samuel</td>
<td>Loyd</td>
</tr>
<tr>
<td>Mikhail</td>
<td>Dobuzhinsky</td>
</tr>
<tr>
<td>James Beach</td>
<td>Wakefield</td>
</tr>
<tr>
<td>Mstislav</td>
<td>Dobuzhinsky</td>
</tr>
<tr>
<td>James</td>
<td>Wakefield</td>
</tr>
</tbody>
</table>

- **Unorganized set of names**
Key Insight

Learn name phylogenies

William Ronald Dodds Fairbairn

Ronald Fairbairn

• Ideal: matched name pairs

Ronald Fairbairn

W. R. D. Fairbairn

• Sets of matching names

John Wilkins

Samuel Loyd

Mstislav Dobuzhinsky

Mikhail Dobuzhinsky

James Beach Wakefield

James Wakefield

• Unorganized set of names
WHY A NAME PHYLOGENY?

- Aligns matching names for transducer
- Organizes names into connected components (clusters)
- We can jointly estimate a phylogeny and a mutation model (transducer)
  - A mutation model gives a phylogeny
  - A phylogeny provides training data for a mutation model
OUTLINE

• Generative model
• Inference
• Experiments
GENERATIVE MODEL
NAME VARIATION

• A generative model of strings that can explain observed name variation
  
  Mitt Romney
  President Barack Obama
  Barack Obama
  Secretary of State Hillary Clinton
  Hillary Clinton
  Barack Obama
  Clinton
  Obama

• What are the sources of variation for names?
Suppose an author decides to write a name

Where do names come from?

- Copy a previous mention
- Mutate a previous mention
- According to mutation model
- Create a new name
COPY A PREVIOUS MENTION

- Select a previous mention at random (uniformly)

- Copy it with probability $1 - \mu$
MUTATE PREVIOUS MENTION

• Select a previous mention at random (uniformly)

• Mutate it with probability $\mu$

• Sample a new mutation from the mutation model given the mention
CREATE A NEW NAME

• Select the root of the phylogeny ♦ with probability proportional to $\alpha$

• Sample a new name from a character language model
SUMMARY

• To generate the next mention
  • Pick an existing name mention $w$ with probability $1/(\alpha + k)$
    • Copy $w$ verbatim with probability $1 - \mu$
    • Mutate $w$ with probability $\mu$
  • Decide to talk about a new entity with probability $\alpha/(\alpha + k)$
    • Generate a name for it
INFERENCES
EM ALGORITHM

- **E-step**
  - Given mutation model $\theta$, compute a distribution over phylogenies

- **M-step**
  - Re-estimate $\theta$ given marginal edge probabilities
    - Sum over alignments for all (x,y) string pairs via forward-backward
    - Each pair is training example weighted by the marginal probability
SUMMARY

• Learn a name matching algorithm
  • \( \theta \) (transducer/mutation model)
  • Phylogeny: a means to an end
    • Part of the reason for a distribution over phylogenies
• Question: Is \( \theta \) better than other name matching algorithms?
  • Can \( \theta \) find matching names more accurately?
EXPERIMENTS
DATA

• English Wikipedia (2011) to create lists of name variants
  • Wikipedia redirects are human-curated pages to resolve common name variants to the correct page (unambiguously)
  • Use Freebase to restrict to redirects for Person entities
  • Applied some further filters to remove redirects that were clearly not names (e.g. numbers)
  • Use LDC Gigaword to obtain a frequency for each name variant
Our Algorithm
Our Algorithm $\theta$ (Transducer)
Our Algorithm

θ (Transducer)
EXPERIMENT: RANKING

• Input: query (name)

• Output: ranked list of possible aliases

• Evaluation: where is correct alias in list?
  • Mean Reciprocal Rank (MRR) (higher is better)
SETUP

• Data
  • Train: 1500 entities (~6000 names)
  • Test: 1500 different entities (~6000 names)

• Settings
  • Train $\theta$ on a set of “supervised” pairs (varying levels of training)

• Baselines: other name matching algorithms
Jaro Winkler
Levenshtein
10 entities
10+unlabeled
Unsupervised
1500 entities

MRR

0.611
Jaro Winkler | Levenshtein

MRR

0.611 | 0.642

- 10 entities
- 10+unlabeled
- Unsupervised
- 1500 entities
The diagram compares different methods and datasets for entity similarity retrieval. The methods include Jaro Winkler and Levenshtein, and the datasets range from 10 entities to 1,500 entities, with options for labeled or unlabeled data. The MRR (Mean Reciprocal Rank) values are as follows:

- Jaro Winkler: 0.611
- Levenshtein: 0.642
- 10 entities: 0.741
- 10+unlabeled: Not applicable
- Unsupervised: Not applicable
- 1500 entities: Not applicable

The higher the MRR, the better the performance of the method or dataset in terms of retrieval accuracy.
The diagram shows a comparison of MRR (Mean Reciprocal Rank) for different methods and datasets.

- **Jaro Winkler** has an MRR of 0.611.
- **Levenshtein** has an MRR of 0.642.
- The **10 entities** dataset has an MRR of 0.741.
- The **10+unlabeled** dataset has an MRR of 0.764.
- The **Unsupervised** method on the **1500 entities** dataset has an MRR of 0.763.

The highest MRR is observed for the Unsupervised method on the 1500 entities dataset.
FUTURE WORK

• Include context for full entity disambiguation
• Increase matching speed
• More sophisticated mutation models
  • Incorporate internal name structure
• Informal genres
• Cross lingual data
QUESTIONS

Nicholas Andrews, Jason Eisner, Mark Dredze.

**Name Phylogeny: A Generative Model of String Variation.** *Empirical Methods in Natural Language Processing (EMNLP), 2012.*

---

**Abstract**

Many linguistic and social processes involve an underlying process of variation. We observe how to learn a string-to-string transducer from a collection of strings (e.g., strings of names). This allows us to learn the transducer from the strings by generating a sequence of strings. We define string-to-string transducer as a sequence of strings, where each string is a sequence of tokens. These transducers are often used to capture such relationships. They model a conditional distribution over strings, which is used to generate new strings.

---

**1 Introduction**

Systematic relationships between pairs of strings are at the core of problems such as transduction (Knight and Graehl, 1998), morphological (Dreyer and Eisner, 2011), cross-document correction (Bhattacharyya and Baldwin, 1998), canonicalization (Callo et al., 2009), and part-of-speech tagging (Buntalay and Lee, 2003). Stochastic transducers such as probabilistic finite-state transducers are often used to capture such relationships. They model a conditional distribution over strings, which is used to generate new strings (Dreyer et al., 2008).

In this paper, we are interested in learning from an uncorrupted collection of strings, some of which might have been derived from others by non-derivative linguistic processes such as abbreviation, morphological derivation, historical sound or spelling change, loanword formation, or translation. Existing models assume that each string is derived from at most one parent, but this may not be the case. The difficulty is that in all of these cases of overlap in the collection of strings, there is some overlap in the collection of strings. We assume that each string is derived from at most one parent, but this may not be the case. The difficulty is that in all of these cases of overlap in the collection of strings, there is some overlap in the collection of strings. We assume that each string is derived from at most one parent, but this may not be the case.