Automatic Discovery of Personal Name Aliases from the Web

Presenter: Chen, Zhong-Yong
Authors: Danushka Bollegala, The University of Tokyo, Tokyo
         Yutaka Matsuo, The University of Tokyo, Tokyo
         Mitsuru Ishizuka, The University of Tokyo, Tokyo

TKDE 2011, vol. 23, no. 6
Summarization

- This paper extracted the lexical pattern to find person name's aliases, then used frequency, page-count, and co-occurrence in the anchor text to do aliases' ranking, then used SVM for training to determine the function of ranking, after which it could automatically obtain aliases, then do ranking to select representation aliases. The experimental part证实 can find high-confidence aliases and assist the relation detection and web search task.
Outline

• 1. Introduction
• 2. Related Work
• 3. Method
• 4. Experiments
• 5. Implementation Considerations
• 6. Discussion
• 7. Conclusion
Retrieving information about people from web search engines can become difficult when a person has nicknames or name aliases. For example, a newspaper article on the baseball player might use the real name, Hideki Matsui, whereas a blogger would use the alias, Godzilla, in a blog entry.
Identification of entities on the web is difficult for two fundamental reasons:

1. Different entities can share the same name (i.e., lexical ambiguity).

2. A single entity can be designated by multiple names (i.e., referential ambiguity).
For example, the lexical ambiguity consider the name *Jim Clark*. Aside from the two most popular namesakes, the formula-one racing champion and the founder of Netscape.

Referential ambiguity occurs because people use different names to refer to the same entity on the web.
Introduction (4/7)

• The problem of referential ambiguity of entities on the web has received much less attention.

• In this paper, the authors examine on the problem of automatically extracting the various references on the web of a particular entity.
Introduction (5/7)

• The contributions can be summarized as follows:
• 1. Propose a lexical pattern-based approach to extract aliases of a given name using snippets returned by a web search engine.
  – The lexical patterns are generated automatically using a set of real world name alias data.
  – Evaluate the confidence of extracted lexical patterns and retain the patterns that can accurately discover aliases for various personal names.
2. To select the best aliases among the extracted candidates, the authors propose numerous ranking scores based upon three approaches:
   – lexical pattern frequency
   – word co-occurrences in an anchor text graph
   – page counts on the web.
• 3. Train a ranking support vector machine to learn the optimal combination of individual ranking scores to construct a robust alias extraction method.

• Conduct a series of experiments to evaluate the various components of the proposed method on three data sets:
  – An English personal names data set
  – An English place names data set
  – A Japanese personal names data set.
Related Work (1/6)

• **Alias identification** is closely related to the problem of cross-document coreference resolution in which the objective is to determine whether two mentions of a name in different documents refer to the same entity.
Related Work (2/6)

• Bagga and Baldwin [10] proposed a cross-document coreference resolution algorithm by first performing within document coreference resolution for each individual document to extract coreference chains.

Related Work (3/6)

• In personal name disambiguation the goal is to disambiguate various people that share the same name (namesakes) [3], [4].

Related Work (4/6)

• However, the name disambiguation problem differs fundamentally from that of alias extraction.

• Because in name disambiguation the objective is to identify the different entities that are referred by the same ambiguous name;
  – in alias extraction, the authors are interested in extracting all references to a single entity from the web.
Related Work (5/6)

- Approximate string matching algorithms have been used for extracting variants or abbreviations of personal names [11].

• Bilenko and Mooney [12] proposed a method to learn a string similarity measure to detect duplicates in bibliography databases.

• However, an inherent limitation of such string matching approaches is that they cannot identify aliases.

Web search engine

seed queries: "NAME* ALIAS"

snippets

"NAME PAT*"

snippets and page counts

word co-occurrence statistics

Anchor texts and urls

OUTPUT: Aliases

training data

(NAME, ALIAS)

Pattern extraction algorithm

Candidate alias extraction algorithm

Candidates ranking algorithm

Lexical patterns

Candidate aliases

INPUT: real name
Method

Extracting Lexical Patterns from Snippets

• For names and aliases, snippets convey useful semantic clues that can be used to extract lexical patterns that are frequently used to express aliases of a name.
  – For example, consider the snippet returned by Google for the query “Will Smith * The Fresh Prince.”

• The authors use the wildcard operator * to perform a NEAR query and it matches with one or more words in a snippet.
Method

Extracting Lexical Patterns from Snippets

• In Fig. 2 the snippet contains aka (i.e., also known as), which indicates the fact that fresh prince is an alias for Will Smith.

Fig. 2. A snippet returned for the query “Will Smith * the Fresh Prince” by Google.
Method

Extracting Lexical Patterns from Snippets

• Propose the shallow pattern extraction method illustrated in Fig. 3 to capture the various ways in which information about aliases of names is expressed on the web.

```
Algorithm 3.1: ExtractPatterns(S)

comment: S is a set of (NAME, ALIAS) pairs
P ← null
for each (NAME, ALIAS) ∈ S
  D ← GetSnippets("NAME * ALIAS")
  for each snippet d ∈ D
    do P ← P + CreatePattern(d)
return (P)
```

• Given a set S of (NAME, ALIAS) pairs, the function ExtractPatterns returns a list of lexical patterns that frequently connect names and their aliases in web snippets.
Method

Extracting Lexical Patterns from Snippets

• Propose the shallow pattern extraction method illustrated in Fig. 3 to capture the various ways in which information about aliases of names is expressed on the web.

```
Algorithm 3.1: ExtractPatterns(S)

comment: S is a set of (NAME, ALIAS) pairs
P ← null
for each (NAME, ALIAS) ∈ S
  D ← GetSnippets("NAME * ALIAS")
  do for each snippet d ∈ D
    do P ← P + CreatePattern(d)
return (P)
```

• The GetSnippets function downloads snippets from a web search engine for the query “NAME * ALIAS.”

Fig. 3. Given a set of (NAME, ALIAS) instances, extract lexical patterns.
Method

Extracting Lexical Patterns from Snippets

• Propose the **shallow pattern extraction method** illustrated in Fig. 3 to capture the various ways in which information about aliases of names is expressed on the web.

![Algorithm 3.1: ExtractPatterns(S)](image)

> From each snippet, the **Create-Pattern** function extracts the sequence of words that appear between the name and the alias.

![Fig. 3. Given a set of (NAME, ALIAS) instances, extract lexical patterns.](image)
Method

Extracting Lexical Patterns from Snippets

• The real name and the alias in the snippet are, respectively, replaced by two variables [NAME] and [ALIAS] to create patterns.

• The definition of lexical patterns includes patterns that contain words as well as symbols such as punctuation markers.
Method

Extracting Lexical Patterns from Snippets

• For example, from the snippet shown in Fig. 2, the lexical pattern is \([\text{NAME}], \text{aka } [\text{ALIAS}]\).

---

...Rock the House, the duo’s debut album of 1987, demonstrated that \textbf{Will Smith}, aka \textit{the Fresh Prince}, was an entertaining and amusing storyteller...

---

Fig. 2. A snippet returned for the query “Will Smith * the Fresh Prince” by Google.

• Repeat the process described above for the reversed query, “\textit{ALIAS} * \textbf{NAME}.”
Method

Extracting Lexical Patterns from Snippets

• Limit the number of matched words with “*” to a maximum of five words.

• Use the patterns to extract candidate aliases for a given name.
Method

Extracting Lexical Patterns from Snippets

• Given a name, NAME and a set, $P$ of lexical patterns, the function `ExtractCandidates` returns a list of candidate aliases for the name.

```
Algorithm 3.2: ExtractCandidates(NAME, P)

comment: $P$ is the set of patterns

$C \leftarrow \text{null}$

for each pattern $p \in P$

    $D \leftarrow \text{GetSnippets(“NAME p *”)}$

    do

        for each snippet $d \in D$

            do $C \leftarrow C + \text{GetNgrams}(d, NAME, p)$

return ($C$)
```

Fig. 4. Given a name and a set of lexical patterns, extract candidate aliases.
Method

Extracting Lexical Patterns from Snippets

- Associate the given name with each pattern, $p$ in the set of patterns, $P$ and produce queries of the form: “NAME $p$ * .”

**Algorithm 3.2:** EXTRACTCANDIDATES(NAME, P)

**comment:** $P$ is the set of patterns

$$C \leftarrow null$$

for each pattern $p \in P$

$$D \leftarrow \text{GetSnippets(“NAME $p$ * ”)}$$

do for each snippet $d \in D$

$$do C \leftarrow C + \text{GetNgrams}(d, NAME, p)$$

return $(C)$

Fig. 4. Given a name and a set of lexical patterns, extract candidate aliases.
Method

Extracting Lexical Patterns from Snippets

- The **GetSnippets** function downloads a set of snippets for the query.

---

**Algorithm 3.2: ExtractCandidates(NAME, P)**

```plaintext
comment: P is the set of patterns
C ← null
for each pattern p ∈ P
    D ← GetSnippets("NAME p *")
    do { for each snippet d ∈ D
            do C ← C + GetNgrams(d, NAME, p)
    }
return (C)
```

---

Fig. 4. Given a name and a set of lexical patterns, extract candidate aliases.
Method

Extracting Lexical Patterns from Snippets

- The GetNgrams function extracts continuous sequences of words (n-grams) from the beginning of the part that matches the wildcard operator *.

Algorithm 3.2: EXTRACTCANDIDATES(NAME, P)

comment: $P$ is the set of patterns

$C ← null$

for each pattern $p ∈ P$

\[
D ← \text{GetSnippets(”NAME p * ”)}
\]

for each snippet $d ∈ D$

\[
do C ← C + \text{GetNgrams}(d, NAME, p)
\]

return $(C)$

Fig. 4. Given a name and a set of lexical patterns, extract candidate aliases.
Method

Extracting Lexical Patterns from Snippets

• Selected up to five grams as candidate aliases.
• Removed candidates that contain only stop words such as a, an, and the.

  – For example, assuming that we retrieved the snippet in Fig. 3 for the query “Will Smith aka*,” the procedure described above extracts the fresh and the fresh prince as candidate aliases.

Fig. 2. A snippet returned for the query “Will Smith * the Fresh Prince” by Google.
Method

Extracting Lexical Patterns from Snippets

• Limit the number of snippets downloaded by the function `GetSnippets` to a maximum of 100 in both Algorithm 3.1 and 3.2.
Method

Ranking of Candidates

- The candidates extracted by the lexical patterns might include some invalid aliases.
- The authors model this problem of alias recognition as one of ranking candidates with respect to a given.
- Define various ranking scores to measure the association between a name and a candidate alias using three different approaches:
  - 1. Lexical pattern frequency
  - 2. Word co-occurrences in an anchor text graph
  - 3. Page counts on the web
Method

Lexical Pattern Frequency

• If the personal name under consideration and a candidate alias occur in many lexical patterns, then it can be considered as a good alias for the personal name.

• Consequently, rank a set of candidate aliases in the descending order of the number of different lexical patterns in which they appear with a name.
Method

Co-Occurrences in Anchor Texts

• Anchor texts are particularly attractive because they not only contain concise texts, but also provide links that can be considered as expressing a citation.

• For example, if the majority of inbound anchor texts of a url contain a personal name, it is likely that the remainder of the inbound anchor texts contain information about aliases of the name.
  – Use the term inbound anchor texts to refer the set of anchor texts pointing to the same url.
Method

Co-Occurrences in Anchor Texts

• Define a name $p$ and a candidate alias $x$ as co-occurring, if $p$ and $x$ appear in two different inbound anchor texts of a url $u$.
  – Do not consider co-occurrences of an alias and a name in the same anchor text.
Method

Co-Occurrences in Anchor Texts

• For example, consider the picture of Will Smith shown in Fig. 5.

• Fig. 5 shows a picture of Will Smith being linked to by four different anchor texts.

Fig. 5. A picture of Will Smith being linked by different anchor texts on the web.
Method

Co-Occurrences in Anchor Texts

• Consider all the words in an anchor text and their bigrams as potential candidate aliases if they co-occur with the real name according to our definition.
Method

Co-Occurrences in Anchor Texts

TABLE 1
Contingency Table for a Candidate Alias $x$

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$C - { x }$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>$\frac{k}{V - {p}}$</td>
<td>$\frac{n - k}{K - k}$</td>
<td>$\frac{n}{N - n}$</td>
</tr>
<tr>
<td>$V - {p}$</td>
<td>$\frac{K}{K}$</td>
<td>$\frac{N - n - K + k}{N - K}$</td>
<td>$\frac{n}{N}$</td>
</tr>
</tbody>
</table>

- $C$ is the set of candidates.
Method

Co-Ocurrences in Anchor Texts

TABLE 1
Contingency Table for a Candidate Alias \( x \)

<table>
<thead>
<tr>
<th></th>
<th>( x )</th>
<th>( C - {x} )</th>
<th>( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( k )</td>
<td>( n - k )</td>
<td>( n )</td>
</tr>
<tr>
<td>( V - {p} )</td>
<td>( K - k )</td>
<td>( N - n - K + k )</td>
<td>( N - n )</td>
</tr>
<tr>
<td>( V )</td>
<td>( K )</td>
<td>( N - K )</td>
<td>( N )</td>
</tr>
</tbody>
</table>

- \( V \) is the set of all words that appear in anchor texts.
Method

Co-Occurrences in Anchor Texts

\[
\begin{array}{|c|c|c|}
\hline
\text{ } & \text{x} & \text{C \setminus \{x\}} \\
\hline
\text{p} & \text{k} & \text{n-k} \\
\hline
\frac{p}{V - \{p\}} & \text{K-k} & \text{N-n-K+k} \\
\hline
\frac{V - \{p\}}{V} & \text{K} & \text{N-K} \\
\hline
\text{C} & \text{n} & \text{N-n} \\
\hline
\end{array}
\]

• \( k \) is the co-occurrence frequency between \( x \) and \( p \).
Method

Co-Occurrences in Anchor Texts

• $K$ is the sum of co-occurrence frequencies between $x$ and all words in $V$.

• $n$ is the sum of co-occurrence frequencies between $p$ and all candidates in $C$.

• $N$ is the total co-occurrences between all word pairs taken from $C$ and $V$.

TABLE 1
Contingency Table for a Candidate Alias $x$

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$C - {x}$</th>
<th>$C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V - {p}$</td>
<td>$k$</td>
<td>$n - k$</td>
<td>$n$</td>
</tr>
<tr>
<td>$V$</td>
<td>$K - k$</td>
<td>$N - n - K + k$</td>
<td>$N - n$</td>
</tr>
</tbody>
</table>
Method

Co-Occurrences in Anchor Texts

• To measure the strength of association between a name and a candidate alias, using Table 1, the authors define nine popular co-occurrence statistics.
Method

Co-Occurrences in Anchor Texts

CF (Co-occurring Frequency)

• The value $k$ in Table 1 denotes the $CF$ of a candidate alias $x$ and a name $p$.

• If there are many urls, which are pointed to by anchor texts that contain a candidate alias $x$ and a name $p$, then it is an indication that $x$ is indeed a correct alias of the name $p$. 
Method

Co-Occurrences in Anchor Texts

**tfidf**

- The *tfidf* score of a candidate *x* as an alias of name *p*; $tfidf(p, x)$ is computed from Table 1 as:

$$tfidf(p, x) = k \log \left( \frac{N}{K + 1} \right)$$

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><em>p</em></td>
<td><em>x</em></td>
<td>$C - {x}$</td>
<td>$C$</td>
</tr>
<tr>
<td>$V - {p}$</td>
<td>$\frac{k}{K}$</td>
<td>$n - k$</td>
<td>$n$</td>
</tr>
<tr>
<td>$V$</td>
<td>$\frac{K - k}{K}$</td>
<td>$N - n - K + k$</td>
<td>$N - n$</td>
</tr>
</tbody>
</table>

TABLE 1
Contingency Table for a Candidate Alias *x*
Method

Co-Occurrences in Anchor Texts
CS(Chi-Squared measure)

• Define $x^2$ ranking score, $CS(p, x)$ of a candidate $x$ as an alias of a name $p$ as follows:

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}.$$  \hspace{1cm} (2)

$$CS(p, x) = \frac{N\{k(N-K-n+k) - (n-k)(K-k)\}^2}{nK(N-K)(N-n)}.$$  \hspace{1cm} (3)

$$CS(p, x) = \left( \frac{k}{N} - \left( \frac{K}{N} \right) \left( \frac{n}{N} \right) \right)^2 + \ldots$$
Method
Co-Occurrences in Anchor Texts
LLR(Log-Likelihood Ratio)

- Likelihood ratios are robust against sparse data and have a more intuitive definition.

- The LLR-based alias ranking score $\text{LLR}(p, x)$ is computed using values in Table 1 as:

$$
\text{LLR}(p, x) = k \log \frac{kN}{nK} + (n - k) \log \frac{(n - k)N}{n(N - K)} \\
+ (K - k) \log \frac{N(K - k)}{K(N - n)} \\
+ (N - K - n + k) \log \frac{N(N - K - n + k)}{(N - K)(N - n)}
$$

<table>
<thead>
<tr>
<th>$V - {p}$</th>
<th>$p$</th>
<th>$V$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$k$</td>
<td>$K - k$</td>
</tr>
<tr>
<td>$C - {x}$</td>
<td>$n - k$</td>
<td>$N - n - K + k$</td>
</tr>
<tr>
<td>$C$</td>
<td>$n$</td>
<td>$N - n$</td>
</tr>
</tbody>
</table>

**TABLE 1**
Contingency Table for a Candidate Alias $x$
Method

Co-Occurrences in Anchor Texts

PMI (Pointwise Mutual Information)

• For values of two random variables $y$ and $z$, their PMI is defined as Formula 5:

$$PMI(y, z) = \log_2 \frac{P(y, z)}{P(y)P(z)}.$$ (5)

• The probabilities in Formula 5 can be computed as marginal probabilities from Table 1 as:

$$PMI(p, x) = \log_2 \frac{k}{Kn} = \log_2 \frac{kN}{Kn}.$$ (6)
Method

Co-Occurrences in Anchor Texts

Hypergeometric Distribution

- For example, the probability of the event that “k red balls are contained among n balls, which are arbitrarily chosen from among N balls containing K red balls” is given by the hypergeometric distribution \( hg(N, K, n, k) \) as:

\[
hg(N, K, n, k) = \frac{(K \choose k)(N-K \choose n-k)}{(N \choose n)}. \tag{7}
\]
Method

Co-Occurrences in Anchor Texts

Hypergeometric Distribution

- Apply the definition 7 of hypergeometric distribution to the values in Table 1 and compute the probability $HG(p, x)$ of observing more than $k$ number of co-occurrences of the name $p$ and candidate alias $x$.
- The value of $HG(p, x)$ is given by

$$HG(p, x) = -\log_2 \left( \sum_{l \geq k} h_N(K, n, l) \right),$$

$$\max\{0, N + K - n\} \geq l \geq \min\{n, K\}.$$  

- The value $HG(p, x)$ indicates the significance of co-occurrences between $p$ and $x$. Use $HG(p, x)$ to rank candidate aliases of a name.
Method

Co-Occurrences in Anchor Texts

Cosine

- Define $\text{cosine}(p, x)$ as a measure of association between a name and a candidate alias as:

$$\text{cosine}(p, x) = \frac{k}{\sqrt{n}\sqrt{K}}. \quad (10)$$

\[
\begin{array}{|c|c|c|c|}
\hline
 & x & C - \{x\} & C \\
\hline
 p & \frac{k}{K} & n - k & \frac{n}{N - n} \\
\hline
 V - \{p\} & \frac{K - k}{N - K} & N - n - K + k & N - n \\
\hline
\end{array}
\]
Method

Co-Occurrences in Anchor Texts

Overlap

• Define a ranking score based on the overlap to evaluate the appropriateness of a candidate alias

\[
\text{overlap}(X, Y) = \frac{|X \cap Y|}{\min(|X|, |Y|)}. \quad (11)
\]

\[
\text{overlap}(p, x) = \frac{k}{\min(n, K)}. \quad (12)
\]
Method

Co-Occurrences in Anchor Texts

Dice

• Define a ranking score based on the Dice as:

\[
\text{Dice}(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|}.
\]

(13)

\[
\text{Dice}(p, x) = \frac{2k}{n + K}.
\]

<table>
<thead>
<tr>
<th>V \setminus {p}</th>
<th>x</th>
<th>C \setminus {x}</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>p \quad V</td>
<td>\underline{k}</td>
<td>\underline{n - k}</td>
<td>\underline{n}</td>
</tr>
<tr>
<td>K \quad K</td>
<td>N - n - K + k</td>
<td>N - n</td>
<td>N</td>
</tr>
</tbody>
</table>

TABLE 1
Contingency Table for a Candidate Alias \( x \)
Method

Hub Discounting

• A frequently observed phenomenon related to the web is that many pages with diverse topics link to so-called hubs such as Google, Yahoo, or MSN.
• Two anchor texts might link to a hub for entirely different reasons.
• Therefore, co-occurrences coming from hubs are prone to noise.
Method

Hub Discounting

- If the majority of anchor texts linked to a particular web site use the real name to do so, then the confidence of that page as a source of information regarding the person whom we are interested in extracting aliases increases.

Fig. 6. Discounting the co-occurrences in hubs.
Method

Hub Discounting

- To overcome the adverse effects of a hub $h$ when computing co-occurrence measures, multiply the number of co-occurrences of words linked to $h$ by a factor $\alpha(h, p)$, where

$$\alpha(h, p) = \frac{t}{d}. \quad (14)$$

- $t$ is the number of inbound anchor texts of $h$ that contain the real name $p$.
- $d$ is the total number of inbound anchor texts of $h$. 
Method

Page-Count-Based Association Measures

• Define word association measures that consider co-occurrences not only in anchor texts but in the web overall.

• Page counts retrieved from a web search engine for the conjunctive query, “p and x,” for a name p and a candidate alias x can be regarded as an approximation of their co-occurrences in the web.
Method

Page-Count-Based Association Measures

WebDice

• \( \text{WebDice}(p, x) : \)

$$\text{WebDice}(p, x) = \frac{2 \times \text{hits}("p \text{ AND } x")}{\text{hits}(p) + \text{hits}(x)}. \quad (15)$$

• \( \text{hits}(q) \) is the page counts for the query \( q \).
Method

Page-Count-Based Association Measures

WebPMI

• \( \text{WebPMI}(p, x) \):

\[
\text{WebPMI}(p, x) = \log_2 \frac{L \times \text{hits(“p AND x”)}}{\text{hits}(p) \times \text{hits}(x)}. \tag{16}
\]

• \( L \) is the number of pages indexed by the web search engine, which the authors approximated as \( L = 10^{10} \) according to the number of pages indexed by Google.
Method

Page-Count-Based Association Measures

*Conditional Probability*

\[
Prob(x|p) = \frac{\text{hits(“p AND x”)}}{\text{hits}(p)}.
\]

\[
Prob(p|x) = \frac{\text{hits(“p AND x”)}}{\text{hits}(x)}.
\]
Method

Training

• Using a data set of name-alias pairs, train a ranking support vector machine to rank candidate aliases according to their strength of association with a name.

• For a name-alias pair, we define three types of features:
  – anchor text-based co-occurrence measures
  – web page-count-based association measures
  – frequencies of observed lexical patterns
Method

Training

• The nine co-occurrence measures are computed with and without weighting for hubs to produce 18(2 x 9) features.

• The four page-count-based association measures.

• The frequency of lexical patterns extracted by Algorithm 3.1 are used as features in training the ranking SVM.
Method

Training

• Normalize each measure to range [0,1] to produce feature vectors for training.
• The trained SVM model can then be used to assign a ranking score to each candidate alias.
• Finally, the highest-ranking candidate is selected as the correct alias of the name.
Method

Data Set

• Create three name-alias data sets:
  – The English personal names data set (50 names)
  – The English place names data set (50 names)
  – The Japanese personal names (100 names) data set.
Method

Data Set

• Aliases were manually collected after referring various information sources such as Wikipedia and official home pages.

• A website might use links for purely navigational purposes, which do not convey any semantic clues.

• In order to remove navigational links in the data set, prepare a list of words that are commonly used in navigational menus such as top, last, next, previous, links, etc., and ignore anchor texts that contain these words.
Method

Data Set

• Remove any links that point to pages within the same site.

• Data set contains 24,456,871 anchor texts pointing to 8,023,364 urls.

• All urls in the data set contain at least two inbound anchor texts.

• The average number of anchor texts per url is 3.05 and the standard deviation is 54.02.
Method

Data Set

• Tokenize anchor texts using the Japanese morphological analyzer MeCab [25].

Experiments

Pattern Extraction

- Algorithm 3.1 extracts over 8,000 patterns for the 50 English personal names data set.
- Rank the patterns according to their F-scores to identify the patterns that accurately convey information about aliases.

Precision(s) = \frac{\text{No. of correct aliases retrieved by } s}{\text{No. of total aliases retrieved by } s},

Recall(s) = \frac{\text{No. of correct aliases retrieved by } s}{\text{No. of total aliases in the dataset}}.

F(s) = \frac{2 \times \text{Precision}(s) \times \text{Recall}(s)}{\text{Precision}(s) + \text{Recall}(s)}.

**TABLE 2**

Lexical Patterns with the Highest $F$-scores for English Personal Name Aliases

<table>
<thead>
<tr>
<th>Pattern</th>
<th>$F$-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>* aka [NAME]</td>
<td>0.335</td>
</tr>
<tr>
<td>[NAME] aka *</td>
<td>0.322</td>
</tr>
<tr>
<td>[NAME] better known as *</td>
<td>0.310</td>
</tr>
<tr>
<td>[NAME] alias *</td>
<td>0.286</td>
</tr>
<tr>
<td>[NAME] also known as *</td>
<td>0.281</td>
</tr>
<tr>
<td>* nee [NAME]</td>
<td>0.225</td>
</tr>
<tr>
<td>[NAME] nickname *</td>
<td>0.224</td>
</tr>
<tr>
<td>* whose real name is [NAME]</td>
<td>0.205</td>
</tr>
<tr>
<td>[NAME] aka the *</td>
<td>0.187</td>
</tr>
<tr>
<td>* was born [NAME]</td>
<td>0.153</td>
</tr>
</tbody>
</table>
Experiments

Pattern Extraction

- Table 3 shows the patterns with the highest F-scores extracted using the English location names data set.

<table>
<thead>
<tr>
<th>Patterns</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>[NAME] nickname the *</td>
<td>0.739</td>
</tr>
<tr>
<td>[NAME] is nicknamed the *</td>
<td>0.723</td>
</tr>
<tr>
<td>[NAME] employment nickname *</td>
<td>0.627</td>
</tr>
<tr>
<td>[NAME] state flag or *</td>
<td>0.589</td>
</tr>
<tr>
<td>[NAME] nicknamed the *</td>
<td>0.5567</td>
</tr>
<tr>
<td>[NAME] is called the *</td>
<td>0.3199</td>
</tr>
</tbody>
</table>
Experiments

Pattern Extraction

- From Fig. 7, see that when use more lower ranked patterns the recall improves at the expense of precision.

![Graph showing the relationship between the number of top ranking patterns and precision, recall, and F-score.](image)

Fig. 7. Quality of the top ranked patterns for English personal name aliases.
Experiments

Alias Extraction

- Mean reciprocal rank (MRR) and AP [26] is used to evaluate the different approaches. MRR is defined as follows:

\[
\text{MRR} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{R_i}.
\]

(TAB 4)

Comparison with Baselines and Previous Work

<table>
<thead>
<tr>
<th>Method</th>
<th>MRR</th>
<th>AP</th>
<th>Method</th>
<th>MRR</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (Linear)</td>
<td>0.67</td>
<td>0.66</td>
<td>Prob(p</td>
<td>x)</td>
<td>0.14</td>
</tr>
<tr>
<td>SVM (Quad)</td>
<td>0.65</td>
<td>0.64</td>
<td>CS(h)</td>
<td>0.12</td>
<td>0.06</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>0.61</td>
<td>0.60</td>
<td>CF</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>HK</td>
<td>0.63</td>
<td>0.62</td>
<td>cosine</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>tfidf(h)</td>
<td>0.39</td>
<td>0.15</td>
<td>tfidf</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>WebDice</td>
<td>0.39</td>
<td>0.38</td>
<td>Dice</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>LLR(h)</td>
<td>0.39</td>
<td>0.15</td>
<td>overlap(h)</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>cosine(h)</td>
<td>0.37</td>
<td>0.14</td>
<td>PMI(h)</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>CF(h)</td>
<td>0.37</td>
<td>0.15</td>
<td>LLR</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>HG(h)</td>
<td>0.33</td>
<td>0.12</td>
<td>HG</td>
<td>0.04</td>
<td>0.01</td>
</tr>
<tr>
<td>Dice(h)</td>
<td>0.29</td>
<td>0.12</td>
<td>CS</td>
<td>0.01</td>
<td>0</td>
</tr>
<tr>
<td>Prob(x</td>
<td>p)</td>
<td>0.21</td>
<td>0.21</td>
<td>PMI</td>
<td>0.01</td>
</tr>
<tr>
<td>WebPMI</td>
<td>0.14</td>
<td>0.13</td>
<td>overlap</td>
<td>0.01</td>
<td>0</td>
</tr>
</tbody>
</table>
Experiments

Alias Extraction

• $\text{Rel}(r)$ is a binary valued function that returns one if the candidate at rank $r$ is a correct alias for the name.

• $\text{Pre}(r)$ is the precision at rank $r$.

\[
\text{AP} = \frac{\sum_{r=1}^{k} \text{Pre}(r) \times \text{Rel}(r)}{\text{No of correct aliases}}.
\]  \hspace{1cm} (18)

\[
\text{Pre}(r) = \frac{\text{no. of correct aliases in top } r \text{ candidates}}{r}.
\]  \hspace{1cm} (19)
### Experiments

#### Alias Extraction

Denote the hub-weighted versions of anchor text-based co-occurrence measures by \( h \).

**TABLE 4**

Comparison with Baselines and Previous Work

<table>
<thead>
<tr>
<th>Method</th>
<th>MRR</th>
<th>AP</th>
<th>Method</th>
<th>MRR</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (Linear)</td>
<td>0.67</td>
<td>0.66</td>
<td>Prob((p</td>
<td>x))</td>
<td>0.14</td>
</tr>
<tr>
<td>SVM (Quad)</td>
<td>0.65</td>
<td>0.64</td>
<td>CS(h)</td>
<td>0.12</td>
<td>0.06</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>0.61</td>
<td>0.60</td>
<td>CF</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>HK</td>
<td>0.63</td>
<td>0.62</td>
<td>cosine</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>tfidf(h)</td>
<td>0.39</td>
<td>0.15</td>
<td>tfidf</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>WebDice</td>
<td>0.39</td>
<td>0.38</td>
<td>Dice</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>LLR(h)</td>
<td>0.39</td>
<td>0.15</td>
<td>overlap(h)</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>cosine(h)</td>
<td>0.37</td>
<td>0.14</td>
<td>PMI(h)</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>CF(h)</td>
<td>0.37</td>
<td>0.15</td>
<td>LLR</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>HG(h)</td>
<td>0.33</td>
<td>0.12</td>
<td>HG</td>
<td>0.04</td>
<td>0.01</td>
</tr>
<tr>
<td>Dice(h)</td>
<td>0.29</td>
<td>0.12</td>
<td>CS</td>
<td>0.01</td>
<td>0</td>
</tr>
<tr>
<td>Prob((x</td>
<td>p))</td>
<td>0.21</td>
<td>0.21</td>
<td>PMI</td>
<td>0.01</td>
</tr>
<tr>
<td>WebPMI</td>
<td>0.14</td>
<td>0.13</td>
<td>overlap</td>
<td>0.01</td>
<td>0</td>
</tr>
</tbody>
</table>
Experiments

Alias Extraction

<table>
<thead>
<tr>
<th>Method</th>
<th>MRR</th>
<th>AP</th>
<th>Method</th>
<th>MRR</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (Linear)</td>
<td>0.67</td>
<td>0.66</td>
<td>Prob(p</td>
<td>x)</td>
<td>0.14</td>
</tr>
<tr>
<td>SVM (Quad)</td>
<td>0.65</td>
<td>0.64</td>
<td>CS(h)</td>
<td>0.12</td>
<td>0.06</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>0.61</td>
<td>0.60</td>
<td>CF</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>HK</td>
<td>0.63</td>
<td>0.62</td>
<td>cosine</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>tfidf(h)</td>
<td>0.39</td>
<td>0.15</td>
<td>tfidf</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>WebDice</td>
<td>0.39</td>
<td>0.38</td>
<td>Dice</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>LLR(h)</td>
<td>0.39</td>
<td>0.15</td>
<td>overlap(h)</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>cosine(h)</td>
<td>0.37</td>
<td>0.14</td>
<td>PMI(h)</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>CF(h)</td>
<td>0.37</td>
<td>0.15</td>
<td>LLR</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>HG(h)</td>
<td>0.33</td>
<td>0.12</td>
<td>HG</td>
<td>0.04</td>
<td>0.01</td>
</tr>
<tr>
<td>Dice(h)</td>
<td>0.29</td>
<td>0.12</td>
<td>CS</td>
<td>0.01</td>
<td>0</td>
</tr>
<tr>
<td>Prob(x</td>
<td>p)</td>
<td>0.21</td>
<td>0.21</td>
<td>PMI</td>
<td>0.01</td>
</tr>
<tr>
<td>WebPMI</td>
<td>0.14</td>
<td>0.13</td>
<td>overlap</td>
<td>0.01</td>
<td>0</td>
</tr>
</tbody>
</table>

- Among the numerous individual ranking scores, the best results are reported by the hubweighted tfidf score (tfidf(h)).
- It is noteworthy for anchor text-based ranking scores, the hubweighted version always outperforms the nonhub-weighted counterpart, which justifies the proposed hub-weighting method.
Experiments

Alias Extraction

**TABLE 5**
Overall Performance

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MRR</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>English Personal Names</td>
<td>0.6150</td>
<td>0.6865</td>
</tr>
<tr>
<td>English Place Names</td>
<td>0.8159</td>
<td>0.7819</td>
</tr>
<tr>
<td>Japanese Personal Names</td>
<td>0.6718</td>
<td>0.6646</td>
</tr>
</tbody>
</table>

- With each data set, we performed a **five-fold cross validation**.
- The proposed method reports high scores for both MRR and AP on all three data sets.
Experiments

Alias Extraction

- The proposed method extracts most aliases in the manually created gold standard.

<table>
<thead>
<tr>
<th>Real Name</th>
<th>Extracted Aliases</th>
</tr>
</thead>
<tbody>
<tr>
<td>David Hasselhoff</td>
<td>hoff, michael knight, michael</td>
</tr>
<tr>
<td>Courteney Cox</td>
<td>dirt lucy, lucy, monica</td>
</tr>
<tr>
<td>Al Pacino</td>
<td>michael corleone</td>
</tr>
<tr>
<td>Teri Hatcher</td>
<td>susan mayer, susan, mayer</td>
</tr>
<tr>
<td>Texas</td>
<td>lone star state, lone star, lone</td>
</tr>
<tr>
<td>Vermont</td>
<td>green mountain state, green</td>
</tr>
<tr>
<td>Wyoming</td>
<td>equality state, cowboy state</td>
</tr>
<tr>
<td>Hideki Matsui</td>
<td>Godzilla, nishikori, matsui</td>
</tr>
</tbody>
</table>
Experiments

Alias Extraction

It is noteworthy that most aliases do not share any words with the name nor acronyms, thus would not be correctly extracted from approximate string matching methods.

<table>
<thead>
<tr>
<th>Real Name</th>
<th>Extracted Aliases</th>
</tr>
</thead>
<tbody>
<tr>
<td>David Hasselhoff</td>
<td>hoff, michael knight, michael</td>
</tr>
<tr>
<td>Courteney Cox</td>
<td>dirt lucy, lucy, monica</td>
</tr>
<tr>
<td>Al Pacino</td>
<td>michael corleone</td>
</tr>
<tr>
<td>Teri Hatcher</td>
<td>susan mayer, susan, mayer</td>
</tr>
<tr>
<td>Texas</td>
<td>lone star state, lone star, lone</td>
</tr>
<tr>
<td>Vermont</td>
<td>green mountain state, green,</td>
</tr>
<tr>
<td>Wyoming</td>
<td>equality state, cowboy state</td>
</tr>
<tr>
<td>Hideki Matsui</td>
<td>Godzilla, nishikori, matsu</td>
</tr>
</tbody>
</table>
Experiments

Alias Extraction

It is interesting to see that for actors the extracted aliases include their roles in movies or television dramas (e.g., Michael Knight for David Hasselhoff).

<table>
<thead>
<tr>
<th>Real Name</th>
<th>Extracted Aliases</th>
</tr>
</thead>
<tbody>
<tr>
<td>David Hasselhoff</td>
<td>hoff, michael knight, michael</td>
</tr>
<tr>
<td>Courteney Cox</td>
<td>dirt lucy, lucy, monica</td>
</tr>
<tr>
<td>Al Pacino</td>
<td>michael corleone</td>
</tr>
<tr>
<td>Teri Hatcher</td>
<td>susan mayer, susan, mayer</td>
</tr>
<tr>
<td>Texas</td>
<td>lone star state, lone star, lone</td>
</tr>
<tr>
<td>Vermont</td>
<td>green mountain state, green, equality state, cowboy state</td>
</tr>
<tr>
<td>Wyoming</td>
<td>Godzilla, nishikori, matsui</td>
</tr>
<tr>
<td>Hideki Matsui</td>
<td></td>
</tr>
</tbody>
</table>
Table 7 shows the top three ranking aliases extracted for **Hideki Matsui** by the proposed SVM (linear) measure and the various baseline ranking scores.

<table>
<thead>
<tr>
<th>Method</th>
<th>First</th>
<th>Second</th>
<th>Third</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM (Linear)</td>
<td>Godzilla</td>
<td>Matsui</td>
<td>Yankees</td>
</tr>
<tr>
<td>tfidf(h)</td>
<td>Matsui</td>
<td>hide</td>
<td>Yankees</td>
</tr>
<tr>
<td>LLR(h)</td>
<td>Matsui</td>
<td>hide</td>
<td>Yankees</td>
</tr>
<tr>
<td>cosine(h)</td>
<td>Matsui</td>
<td>Yankees</td>
<td>hide</td>
</tr>
<tr>
<td>CF(h)</td>
<td>Matsui</td>
<td>hide</td>
<td>Yankees</td>
</tr>
<tr>
<td>HG(h)</td>
<td>Matsui</td>
<td>Yankees</td>
<td>hide</td>
</tr>
<tr>
<td>Dice(h)</td>
<td>Matsui</td>
<td>Yankees</td>
<td>hide</td>
</tr>
<tr>
<td>CS(h)</td>
<td>Matsui</td>
<td>Major league</td>
<td>player</td>
</tr>
<tr>
<td>CF</td>
<td>Tokyo</td>
<td>Yomiuri</td>
<td>Nikkei</td>
</tr>
<tr>
<td>cosine</td>
<td>Yomiuri</td>
<td>Tokyo stock exchange</td>
<td>Matsui</td>
</tr>
<tr>
<td>tfidf</td>
<td>Yomiuri</td>
<td>Tokyo stock exchange</td>
<td>Tokyo stock exchange</td>
</tr>
<tr>
<td>Dice</td>
<td>Yomiuri</td>
<td>Tokyo stock exchange</td>
<td>Tokyo stock exchange</td>
</tr>
<tr>
<td>overlap(h)</td>
<td>play</td>
<td>Godzilla</td>
<td>Steinbrenner</td>
</tr>
<tr>
<td>PMI(h)</td>
<td>play</td>
<td>Godzilla</td>
<td>Steinbrenner</td>
</tr>
<tr>
<td>LLR</td>
<td>Yomiuri</td>
<td>Tokyo stock exchange</td>
<td>jiji.com</td>
</tr>
<tr>
<td>HG</td>
<td>Yomiuri</td>
<td>Tokyo stock exchange</td>
<td>Matsui</td>
</tr>
<tr>
<td>CS</td>
<td>jiji.com</td>
<td>Tokyo stock exchange</td>
<td>Yomiuri</td>
</tr>
<tr>
<td>PMI</td>
<td>Komdatzien</td>
<td>picture</td>
<td>contents</td>
</tr>
<tr>
<td>overlap</td>
<td>Komdatzien</td>
<td>picture</td>
<td>contents</td>
</tr>
</tbody>
</table>
The nonhub weighted measures have a tendency to include general terms such as Tokyo, Yomiuri (a popular Japanese newspaper), Nikkei (a Japanese business newspaper), and Tokyo stock exchange.
A close analysis revealed that such general terms frequently co-occur with a name in hubs.
Experiments

Relation Detection

• Evaluate the effect of aliases on a real-world relation detection task as follows:

• Manually classified 50 people in the English personal names data set, depending on their field of expertise, into four categories: music, politics, movies, and sports.
Experiments

Relation Detection

• Measured the association between two people using the PMI (16) between their names on the web

\[
\text{WebPMI}(p, x) = \log_2 \frac{L \times \text{hits}(“p AND x”)}{\text{hits}(p) \times \text{hits}(x)}. \quad (16)
\]
Experiments

Relation Detection

• Use group average agglomerative clustering (GAAC) [18] to group the people into four clusters.

• Correlation $\text{Corr}(\Gamma)$ between two clusters $X$ and $Y$ is defined as

$$\text{Corr}(\Gamma) = \frac{1}{2} \frac{1}{|\Gamma|(|\Gamma| - 1)} \sum_{(u,v) \in \Gamma} \text{sim}(u, v).$$  \hspace{1cm} (20)
Experiments

Relation Detection

• Used the B-CUBED metric [10] to evaluate the clustering results.

• The B-CUBED evaluation metric was originally proposed for evaluating cross-document coreference chains.

Experiments

Relation Detection

• For each person $p$ in the data set, denote the cluster that $p$ belongs to as $C(p)$.

• Use $A(p)$ to represent the affiliation of person $p$, e.g., $A(“Bill Clinton”) = “politics.”$

\[
\text{Precision}(p) = \frac{\text{No. of people in } C(p) \text{ with affiliation } A(p)}{\text{No. of people in } C(p)},
\]

\[
\text{Recall}(p) = \frac{\text{No. of people in } C(p) \text{ with affiliation } A(p)}{\text{TotalNo. of people with affiliation } A(p)}.
\]

\[
F(p) = \frac{2 \times \text{Precision}(p) \times \text{Recall}(p)}{\text{Precision}(p) + \text{Recall}(p)}.
\]

\[
\text{Precision} = \frac{1}{N} \sum_{p \in \text{Data Set}} \text{Precision}(p),
\]

\[
\text{Recall} = \frac{1}{N} \sum_{p \in \text{Data Set}} \text{Recall}(p),
\]

\[
\text{F-Score} = \frac{1}{N} \sum_{p \in \text{Data Set}} F(p).
\]
Experiments

Relation Detection

Table 8 shows that F-scores have increased as a result of including aliases with real names in relation identification.

Moreover, the improvement is largely attributable to the improvement in recall.
Experiments

Web Search Task

• By including an alias that uniquely identifies a person from his or her namesakes, it might be possible to filter out irrelevant search results.
• For a given individual, search Google with the name as the query and collect top 50 search results.
• Manually go through the search results one by one and decide whether they are relevant for the person we searched for.
Experiments

Web Search Task

• Append the name query with an alias of the person and repeat the above-mentioned process.
Experiments

Web Search Task

Table 9 summarizes the experimental results for 10 Japanese names in the data set.

<table>
<thead>
<tr>
<th>No</th>
<th>First Name</th>
<th>Last Name</th>
<th>Alias</th>
<th>F</th>
<th>L</th>
<th>A</th>
<th>F+A</th>
<th>L+A</th>
<th>F+L</th>
<th>F+L+A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hideki</td>
<td>Matsui</td>
<td>godzilla</td>
<td>48</td>
<td>1</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Minako</td>
<td>Nakano</td>
<td>nakami</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>50</td>
<td>38</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Maki</td>
<td>Goto</td>
<td>gomaki</td>
<td>16</td>
<td>6</td>
<td>24</td>
<td>48</td>
<td>44</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Hidetoshi</td>
<td>Nakata</td>
<td>hide</td>
<td>44</td>
<td>6</td>
<td>12</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Takuya</td>
<td>Kimura</td>
<td>kimutaku</td>
<td>34</td>
<td>2</td>
<td>30</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Yuichi</td>
<td>Kimura</td>
<td>brother kimu</td>
<td>29</td>
<td>0</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>Takafumi</td>
<td>Horie</td>
<td>horiemon</td>
<td>29</td>
<td>5</td>
<td>49</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>Hikaru</td>
<td>Utada</td>
<td>hikki</td>
<td>30</td>
<td>48</td>
<td>5</td>
<td>50</td>
<td>48</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>Ryuichi</td>
<td>Sakamoto</td>
<td>professor</td>
<td>22</td>
<td>3</td>
<td>1</td>
<td>36</td>
<td>17</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>10</td>
<td>Miki</td>
<td>Ando</td>
<td>mikiti</td>
<td>41</td>
<td>8</td>
<td>4</td>
<td>47</td>
<td>35</td>
<td>48</td>
<td>50</td>
</tr>
</tbody>
</table>
Experiments

Web Search Task

TABLE 9
Effect of Aliases in Personal Name Search

<table>
<thead>
<tr>
<th>No</th>
<th>First Name</th>
<th>Last Name</th>
<th>Alias</th>
<th>F</th>
<th>L</th>
<th>A</th>
<th>F+A</th>
<th>L+A</th>
<th>F+L</th>
<th>F+L+A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hideki</td>
<td>Matsui</td>
<td>godzilla</td>
<td>48</td>
<td>1</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Minako</td>
<td>Nakano</td>
<td>nakami</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>50</td>
<td>38</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Maki</td>
<td>Goto</td>
<td>gomaki</td>
<td>16</td>
<td>6</td>
<td>24</td>
<td>48</td>
<td>44</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Hidetoshi</td>
<td>Nakata</td>
<td>hide</td>
<td>44</td>
<td>6</td>
<td>12</td>
<td>48</td>
<td>44</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Takuya</td>
<td>Kimura</td>
<td>kimutaku</td>
<td>34</td>
<td>2</td>
<td>30</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Yuichi</td>
<td>Kimura</td>
<td>brother kimu</td>
<td>29</td>
<td>0</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>Takafumi</td>
<td>Horie</td>
<td>horiemon</td>
<td>29</td>
<td>5</td>
<td>49</td>
<td>48</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>Hikaru</td>
<td>Utada</td>
<td>hikki</td>
<td>30</td>
<td>48</td>
<td>5</td>
<td>50</td>
<td>48</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>Ryuichi</td>
<td>Sakamoto</td>
<td>professor</td>
<td>22</td>
<td>3</td>
<td>1</td>
<td>36</td>
<td>17</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>10</td>
<td>Miki</td>
<td>Ando</td>
<td>mikitiki</td>
<td>41</td>
<td>8</td>
<td>4</td>
<td>47</td>
<td>35</td>
<td>48</td>
<td>50</td>
</tr>
</tbody>
</table>

- The number of relevant results have improved for both first name (F) and last name (L) only queries when the aliases was added to the query (F versus F+A and L versus L+A).
Experiments

Web Search Task

TABLE 9
Effect of Aliases in Personal Name Search

<table>
<thead>
<tr>
<th>No</th>
<th>First Name</th>
<th>Last Name</th>
<th>Alias</th>
<th>F</th>
<th>L</th>
<th>A</th>
<th>F+A</th>
<th>L+A</th>
<th>F+L</th>
<th>F+L+A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hideki</td>
<td>Matsui</td>
<td>godzilla</td>
<td>48</td>
<td>1</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Minako</td>
<td>Nakano</td>
<td>nakami</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>50</td>
<td>38</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Maki</td>
<td>Goto</td>
<td>gomaki</td>
<td>16</td>
<td>6</td>
<td>24</td>
<td>48</td>
<td>44</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Hidetoshi</td>
<td>Nakata</td>
<td>hide</td>
<td>44</td>
<td>6</td>
<td>12</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Takuya</td>
<td>Kimura</td>
<td>kimutaku</td>
<td>34</td>
<td>2</td>
<td>30</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Yuichi</td>
<td>Kimura</td>
<td>brother kimu</td>
<td>29</td>
<td>0</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>Takafumi</td>
<td>Horie</td>
<td>horiemon</td>
<td>29</td>
<td>5</td>
<td>49</td>
<td>48</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>Hikaru</td>
<td>Utada</td>
<td>hikki</td>
<td>30</td>
<td>48</td>
<td>5</td>
<td>50</td>
<td>48</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>Ryuichi</td>
<td>Sakamoto</td>
<td>professor</td>
<td>22</td>
<td>3</td>
<td>1</td>
<td>36</td>
<td>17</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>10</td>
<td>Miki</td>
<td>Ando</td>
<td>mikiti</td>
<td>41</td>
<td>8</td>
<td>4</td>
<td>47</td>
<td>35</td>
<td>48</td>
<td>50</td>
</tr>
</tbody>
</table>

• In particular, last names alone produce very poor results. This is because most Japanese last names are highly ambiguous.
Experiments

Web Search Task

TABLE 9
Effect of Aliases in Personal Name Search

<table>
<thead>
<tr>
<th>No</th>
<th>First Name</th>
<th>Last Name</th>
<th>Alias</th>
<th>F</th>
<th>L</th>
<th>A</th>
<th>F+A</th>
<th>L+A</th>
<th>F+L</th>
<th>F+L+A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hideki</td>
<td>Matsui</td>
<td>godzilla</td>
<td>48</td>
<td>1</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Minako</td>
<td>Nakano</td>
<td>nakami</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>50</td>
<td>38</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Maki</td>
<td>Goto</td>
<td>gomaki</td>
<td>16</td>
<td>6</td>
<td>24</td>
<td>48</td>
<td>44</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Hidetoshi</td>
<td>Nakata</td>
<td>hide</td>
<td>44</td>
<td>6</td>
<td>12</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Takuya</td>
<td>Kimura</td>
<td>kimutaku</td>
<td>34</td>
<td>2</td>
<td>30</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Yuichi</td>
<td>Kimura</td>
<td>brother kimu</td>
<td>29</td>
<td>0</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>TakaFumi</td>
<td>Horie</td>
<td>horiemon</td>
<td>29</td>
<td>5</td>
<td>49</td>
<td>48</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>Hikaru</td>
<td>Utada</td>
<td>hikki</td>
<td>30</td>
<td>48</td>
<td>5</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>Ryuichi</td>
<td>Sakamoto</td>
<td>professor</td>
<td>22</td>
<td>3</td>
<td>1</td>
<td>36</td>
<td>17</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>10</td>
<td>Miki</td>
<td>Ando</td>
<td>mikiti</td>
<td>41</td>
<td>8</td>
<td>4</td>
<td>47</td>
<td>35</td>
<td>48</td>
<td>50</td>
</tr>
</tbody>
</table>

• For example, the last name Nakano (person no. 2) is a place name as well as a person name and does not provide any results for Minako Nakano, the television announcer.
Experiments

Web Search Task

• See that searching by the full name (F+L) returns perfect results for seven out of the 10 people.
• However, it is noteworthy that including the aliases still improve relevancy even in the remaining three cases.

<table>
<thead>
<tr>
<th>No</th>
<th>First Name</th>
<th>Last Name</th>
<th>Alias</th>
<th>F</th>
<th>L</th>
<th>A</th>
<th>F+A</th>
<th>L+A</th>
<th>F+L</th>
<th>F+L+A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hideki</td>
<td>Matsui</td>
<td>godzilla</td>
<td>48</td>
<td>1</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Minako</td>
<td>Nakano</td>
<td>nakami</td>
<td>8</td>
<td>0</td>
<td>4</td>
<td>50</td>
<td>38</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Maki</td>
<td>Goto</td>
<td>gomaki</td>
<td>16</td>
<td>6</td>
<td>24</td>
<td>48</td>
<td>44</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Hidetoshi</td>
<td>Nakata</td>
<td>hide</td>
<td>44</td>
<td>6</td>
<td>12</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Takuya</td>
<td>Kimura</td>
<td>kimutaku</td>
<td>34</td>
<td>2</td>
<td>30</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Yuichi</td>
<td>Kimura</td>
<td>brother kimu</td>
<td>29</td>
<td>0</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>Takafumi</td>
<td>Horie</td>
<td>horiemon</td>
<td>29</td>
<td>5</td>
<td>49</td>
<td>48</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>Hikaru</td>
<td>Utada</td>
<td>hikki</td>
<td>30</td>
<td>48</td>
<td>5</td>
<td>50</td>
<td>48</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>Ryuichi</td>
<td>Sakamoto</td>
<td>professor</td>
<td>22</td>
<td>3</td>
<td>1</td>
<td>36</td>
<td>17</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>10</td>
<td>Miki</td>
<td>Ando</td>
<td>mikiti</td>
<td>41</td>
<td>8</td>
<td>4</td>
<td>47</td>
<td>35</td>
<td>48</td>
<td>50</td>
</tr>
</tbody>
</table>
Experiments

Web Search Task

• The time taken to process 100 names in the Japanese personal names is as follows:
  – pattern extraction ca. 3 m (processing only top 100 snippets)
  – candidate alias extraction ca. 5.5 h (using 200 patterns), feature generation 2.4 h
  – training ca. 5 m, and testing 30 s (50 names).

• Overall it takes 8.1 h to run the system end to end. However, it is noteworthy that once the system is trained, detecting an alias of a given name requires only ca. 4.3 m (candidate alias extraction ca. 3.1 m, feature generation 1 m, ranking 10 s).
Discussion

- Lexical patterns can only be matched within the same document.
- In contrast, anchor texts can be used to identify aliases of names across documents.
- The use of **lexical patterns** and **anchor texts**, respectively, can be considered as an approximation of within document and cross-document alias references.
Discussion

• In Section 4.2, the authors showed that by combining both lexical patterns-based features and anchor text-based features, it can achieve better performance in alias extraction.
Discussion

• In Section 4.4, the authors showed experimentally that the knowledge of aliases is helpful to identify a particular person from his or her namesakes on the web.

• Aliases are one of the many attributes of a person that can be useful to identify that person on the web.
Conclusion

• Proposed a lexical-pattern-based approach to extract aliases of a given name.
• The candidates are ranked using various ranking scores computed using three approaches:
  – lexical pattern frequency
  – co-occurrences in anchor texts
  – page counts-based association measures
Conclusion

• Construct a single ranking function using ranking support vector machines.
• The proposed method reported high MRR and AP scores on all three data sets and outperformed numerous baselines and a previously proposed alias extraction algorithm.
• Discounting co-occurrences from hubs is important to filter the noise in co-occurrences in anchor texts.
Conclusion

• The extracted aliases significantly improved recall in a relation detection task and render useful in a web search task.