ijecs
open access www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume - 3 Issue - 8 August, 2014 Page No. 7578-7582

Automatic Discovery of Personal Name Aliases from the Web Using Lexical Pattern-Based Approach

Ms. Trupti M. Marawar , Prof. Wyomesh Deepanker , Prof. Kailash Patel

RGPV University, Technocrats Institute Of Technology(Excellence)
Information Technology Dept.
Bhopal, India
trupti.marawar@rediffmail.com

RGPV University, Technocrats Institute Of Technology(Excellence)
Information Technology Dept
Assistant Professor, Information Technology Dept.
Bhopal, India
wyomeshd@yahoo.com

RGPV University, Technocrats Institute Of Technology(Excellence)
Information Technology Dept
Assistant Professor, Information Technology Dept.
Bhopal, India
patelkailashkumar@gmail.com

Abstract: ***An individual is typically referred by numerous name aliases on the web. Accurate identification of aliases of a given person name is useful in various web related tasks such as information retrieval, sentiment analysis, personal name disambiguation, and relation extraction. We propose a method to extract aliases of a given personal name from the web. Given a personal name, the proposed method first extracts a set of candidate aliases. Second, we rank the extracted candidates according to the likelihood of a candidate being a correct alias of the given name. We propose a novel, automatically extracted lexical pattern-based approach to efficiently extract a large set of candidate aliases from snippets retrieved from a web search engine. We define numerous ranking scores to evaluate candidate aliases using three approaches: lexical pattern frequency, word co occurrences in an anchor text graph, and page counts on the web. To construct a robust alias detection system, we integrate the different ranking scores into a single ranking function using ranking support vector machines.***

Keywords: Web mining, information extraction, web text analysis.

1. Introduction

Searching for information about people in the web is one of the most common activities of internet users. Around 30 percent of search engine queries include person names [1], [2]. However, retrieving information about people from web search engines can become difficult when a person has nicknames or name aliases. For example, the famous Japanese major league baseball player Hideki Matsui is often called as Godzilla on the web. 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. We will not be able to retrieve all the information about the baseball player, if we only use his real name. Identification of entities on the web is difficult for two fundamental reasons: first, different entities can share the same name (i.e., lexical ambiguity); second, a single entity can be designated bymultiple 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, at least 10 different people are listed among the top 100 results returned by Google for the name. On the other hand, referential ambiguity occurs because people use different names to refer to the same entity on the web. For example, the American movie star Will Smith is often

called the Fresh Prince in web contents. Although lexical ambiguity, particularly ambiguity related to personal names has been explored extensively in the previous studies of name disambiguation [3], [4], the problem of referential ambiguity of entities on the web has received much less attention. In this paper, we specifically examine on the problem of automatically extracting the various references on the web of a particular entity. For an entity e, we define the set A of its aliases to be the set of all words or multiword expressions that are used to refer to e on the web. For example, Godzilla is a one-word alias for Hideki Matsui, whereas alias the Fresh Prince contains three words and refers to Will Smith. Various types of terms are used as aliases on the web. For instance, in the case of an actor, the name of a role or the title of a drama (or a movie) can later become an alias for the person (e.g., Fresh Prince, Knight Rider). Titles or professions such as president, doctor, professor, etc., are also frequently used as aliases. Variants or abbreviationsof names such as Bill for William, and acronyms such as JFK for John Fitzgerald Kennedy are also types of name aliases that are observed frequently on the web.

2. Related work

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. 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, and then, clustering the coreference chains under a vector space model to identify all mentions of a name in the document set. However, the vastly numerous documents on the web render it impractical to perform within document coreference resolution to each document separately, and then, cluster the documents to find aliases. In personal name disambiguation the goal is to disambiguate various people that share the same name (namesakes) [3], [4]. Given an ambiguous name, most name disambiguation algorithms have modeled the problem as one of document clustering in which all documents that discuss a particular individual of the given ambiguous name are grouped into a single cluster. The web people search task (WePS)1 provided an evaluation data set and compared various name disambiguation systems. 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, we are interested in extracting all references to a single entity from the web. Approximate string matching algorithms have been used for extracting variants or abbreviations of personal names (e.g., matching Will Smith with the first name initialized variant W. Smith) [11]. Rules in the form of regular expressions and edit-distance-based methods have been used to compare names. 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, which share no words or letters with the real name. For example, approximate string matching methods would not identify Fresh Prince as an alias for Will Smith.

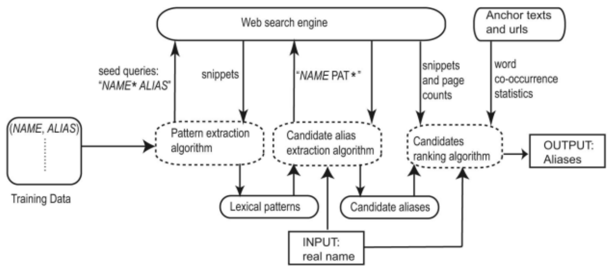

Figure. 1. Outline of the proposed method.

3. Challenges

Identification of entities on the web is difficult for two fundamental reasons: first, different entities can share the same name (i.e., lexical ambiguity); second, 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, at least 10 different people are listed among the top 100 results returned by Google for

the name. On the other hand, referential ambiguity occurs because people use different names to refer to the same entity on the web. For example, the American movie star Will Smith is often called the Fresh Prince in web contents. Although lexical ambiguity, particularly ambiguity related to personal names has been explored extensively in the previous studies of name disambiguation [3], [4], the problem of referential ambiguity of entities on the web has received much less attention.

4. Applications

It can be useful tool for analyzing the name of person from his alias name. In this it can be predictable step for detection of any kind of cyber crime.

5. Analysis of Problem

Identifying aliases of a name are important in information retrieval [5]. In information retrieval, to improve recall of a web search on a person name, a search engine can automatically expand a query using aliases of the name [6]. In our previous example, a user who searches for Hideki Matsui might also be interested in retrieving documents in which Matsui is referred to as Godzilla. Consequently, we can expand a query on Hideki Matsui using his alias name Godzilla. The semantic web is intended to solve the entity disambiguation problem by providing a mechanism to add semantic metadata for entities. However, an issue that the semantic web currently faces is that insufficient semantically annotated web contents are available. Automatic extraction of metadata [7] can accelerate the process of semantic annotation. For named entities, automatically extracted aliases can serve as a useful source of metadata, thereby providing a means to disambiguate an entity.

6. Methods

The proposed method is comprises two main components: pattern extraction, alias extraction and ranking. Using a seed list of name-alias pairs, we first extract lexical patterns that are frequently used to convey information related to aliases on the web. The extracted patterns are then used to find candidate aliases for a given name. We define various ranking scores using the hyperlink structure on the web and page counts retrieved from a search engine to identify the correct aliases among the extracted candidate.

6.1 Extracting Lexical Patterns from Snippets

Many modern search engines provide a brief text snippet for each search result by selecting the text that appears in the web page in the proximity of the query. Such snippets provide valuable information related to the local context of the query. 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 Google2 for the query "Will Smith _ The Fresh Prince."
Here, we use the wildcard operator _ to perform a NEAR query and it matches with one or more words in a snippet.
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. In addition to aka, numerous clues exist such as nicknamed, alias, real name is nee, which are used on the web to representaliases of a name. Consequently, we 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. Lexico-syntactic patterns have been used in numerous related tasks such as extracting hypernyms [14] and meronyms [15].
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. For each (NAME, ALIAS) pair in S, the GetSnippets function downloads snippets from a web search engine for the query "NAME _ ALIAS." Then, from each snippet, the Create- Pattern function extracts the sequence of words that appear between the name and the alias. Results of our preliminary experiments demonstrated that consideration of words that fall outside the name and the alias in snippets did not improve performance. Finally, the real name and the alais in the snippet are, respectively, replaced by two variables [NAME] and [ALIAS] to create patterns.

...Rock the House, the duo's debut album of 1987, demonstrated that **Will Smith**, aka **the Fresh Prince**, was an entertaining and amusing storyteller...

Figure. 2. A snippet returned for the query "Will Smith _ the Fresh Prince" by Google.

```
Algorithm 3.1: EXTRACTPATTERNS($S$)
  comment: $S$ is a set of (NAME, ALIAS) pairs
  $P \leftarrow null$
  for each $(NAME, ALIAS) \in S$
     do { $D \leftarrow$ GetSnippets("$NAME * ALIAS$")
        { for each snippet $d \in D$
        {    do $P \leftarrow P +$ CreatePattern($d$)
  return ($P$)
```

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

```
Algorithm 3.2: EXTRACTCANDIDATES($NAME, P$)
  comment: $P$ is the set of patterns
  $C \leftarrow null$
  for each pattern $p \in P$
     do { $D \leftarrow$ GetSnippets("$NAME\ p\ *$")
        { for each snippet $d \in D$
        {    do $C \leftarrow C +$ GetNgrams($d, NAME, p$)
  return ($C$)
```

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

Our definition of lexical patterns includes patterns that contain words as well as symbols such as punctuation markers. For example, from the snippet shown in Fig. 2, we extract the pattern [NAME], aka [ALIAS]. We repeat the process described above for the reversed query, "ALIAS _ NAME" to extract patterns in which the alias precedes the name. In our experiments, we limit the number of matched words with "_" to a maximum of five words. Because snippets returned by web search engines are very short in length compared to the corresponding source documents, increasing the matching window beyond five words did not produce any additional lexical patterns.
Once a set of lexical patterns is extracted, we use the patterns to extract candidate aliases for a given name as portrayed in Fig. 4. Given a name, NAME and a set, P of lexical patterns, the function ExtractCandidates returns a list of candidate aliases for the name. We associate the given name with each pattern, p in the set of patterns, P and produce queries of the form: "NAME p_." Then, the GetSnippets function downloads a set of snippets for the query. Finally, the GetNgrams function extracts continuous sequences of words (n-grams) from the beginning of the part that matches the wildcard operator _ Experimentally, we selected up to five grams as candidate aliases. Moreover, we 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. For efficiency reasons, we limit the number of snippets downloaded by the function Get Snippets to a maximum of 100 in both Algorithm 3.1 and 3.2. In Google it is possible to retrieve 100 snippets by issuing only a single query by setting the search parameter num to 100, thereby reducing the number queries required in practice.

6.2 Ranking of Candidates

Considering the noise in web snippets, candidates extracted by the shallow lexical patterns might include some invalid aliases. From among these candidates, we must identify those, which are most likely to be correct aliases of a given name. We model this problem of alias recognition as one of ranking candidates with respect to a given name such that the candidates, who are most likely
to be correct aliases are assigned a higher rank.

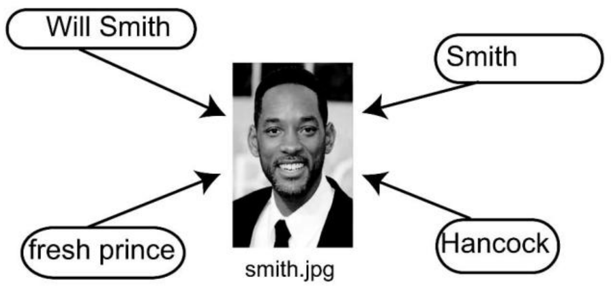

Figure. 5. A picture of Will Smith being linked by different anchor texts on the web.

First, we define various ranking scores to measure the association between a name and a candidate alias using three different approaches: lexical pattern frequency, word co-occurrences in an anchor text graph, and page counts on the web. Next, we describe the those three approaches in detail.

6.3 Lexical Pattern Frequency

In Section 8.1 we presented an algorithm to extract numerous lexical patterns that are used to describe aliases of a personal name. As we will see later in Section 4, the proposed pattern extraction algorithm can extract a large number of lexical patterns. 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, we rank a set of candidate aliases in the descending order of the number of different lexical patterns in which they appear with a name. The lexical pattern frequency of an alias is analogous to the document frequency (DF) popularly used in information retrieval.

6.4 Co-Occurrences in Anchor Texts

Anchor texts have been studied extensively in information retrieval and have been used in various tasks such as synonym extraction, query translation in cross-language information retrieval, and ranking and classification of web pages [16]. Anchor texts are particularly attractive because they not only contain concise texts, but also provide links that can be considered as expressing a citation. We revisit anchor texts to measure the association between a name and its aliases on the web. Anchor texts pointing to a url provide useful semantic clues related to the resource represented by the url. 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. Here, we use the term inbound anchor texts to refer the set of anchor texts pointing to the same url.

7. Proposed Work and Objectives

We propose a fully automatic method to discover aliases of a given personal name from the web. Our contributions can be summarized as follows:

We 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. We evaluate the confidence of extracted lexical patterns and retain the patterns that can accurately discover aliases for various personal names. Our pattern extraction algorithm does not assume any language specific preprocessing such as part-of-speech tagging or dependency parsing, etc., which can be both inaccurate and computationally costly in web-scale data processing. To select the best aliases among the extracted candidates, we propose numerous ranking scores based upon three approaches: lexical pattern frequency, word co-occurrences in an anchor text graph, and page counts on the web. Moreover, using real-world name alias data, we train a ranking support vector machine to learn the optimal combination of individual ranking scores to construct a robust alias extraction method. We conduct a series of experiments to evaluate the various components of the proposed method. We compare the proposed method against numerous baselines and previously proposed name alias extraction methods on three data sets: an English personal names data set, an English place names data set, and a Japanese personal names data set. Moreover, we evaluate the aliases extracted by the proposed method in an information retrieval task and a relation extraction task.

8. Expected Outcome

We utilized both lexical patterns extracted from snippets retrieved from a web search engine as well as anchor texts and links in a web crawl. 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. We showed that by combining both lexical patterns based features and anchor text-based features, we can achieve better performance in alias extraction. It can only incorporate first order co-occurrences. An alias might not always uniquely identify a person. For example, the alias Bill is used to refer many individuals who has the first name William. The namesake disambiguation problem focuses on identifying the different individuals who have the same name. The existing namesake disambiguation algorithms assume the real name of a person to be given and does not attempt to disambiguate people who are referred only by aliases. 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. Extracting common attributes such as date of birth, affiliation, occupation, and nationality have been shown to be useful for namesake disambiguation on the web.

9. Conclusion

We proposed a lexical-pattern-based approach to extract aliases of a given name. We use a set of names and theiraliases as training data to extract lexical patterns that describe numerous ways in which information related to aliases of a name is presented on the web. Next, we substitute the real name of the person that we are interested in finding aliases in the extracted lexical patterns, and download snippets from a web search engine. We extract a set of candidate aliases from the snippets. The candidates are ranked using various ranking scores computed using three approaches: lexical pattern frequency, co-occurrences in anchor texts, and page counts-based association measures.

Moreover, we integrate the different ranking scores to construct a single ranking function using ranking support vector machines.

References

[1] R. Guha and A. Garg, “Disambiguating People in Search”, technical report, Stanford Univ., 2004.

[2] J. Artiles, J. Gonzalo, and F. Verdejo, “A Testbed for People Searching Strategies in the WWW*”,* Proc. SIGIR ’05, pp. 569-570, 2005.

[3] G. Mann and D. Yarowsky, “Unsupervised Personal Name Disambiguation”, Proc. Conf. Computational Natural Language Learning (CoNLL ’03), pp. 33-40, 2003.

[4] R. Bekkerman and A. McCallum, “Disambiguating Web Appearances of People in a Social Network,” Proc. Int’l World Wide Web Conf. (WWW ’05), pp. 463-470, 2005.

[5] G. Salton and M. McGill, Introduction to Modern Information Retreival. McGraw-Hill Inc., 1986.

[6] M. Mitra, A. Singhal, and C. Buckley, “Improving Automatic Query Expansion,” Proc. SIGIR ’98, pp. 206-214, 1998.

[7] P. Cimano, S. Handschuh, and S. Staab, “Towards the Self- Annotating Web,” Proc. Int’l World Wide Web Conf. (WWW ’04), 2004.

[8] Y. Matsuo, J. Mori, M. Hamasaki, K. Ishida, T. Nishimura, H. Takeda, K. Hasida, and M. Ishizuka, “Polyphonet: An Advanced Social Network Extraction System,” Proc. WWW ’06, 2006.

[9] P. Turney, “Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews,” Proc. Assoc. for Computational Linguistics (ACL ’02), pp. 417-424, 2002.

[10] A. Bagga and B. Baldwin, “Entity-Based Cross-Document Coreferencing Using the Vector Space Model,” Proc. Int’l Conf. Computational Linguistics (COLING ’98), pp. 79-85, 1998.

[11] C. Galvez and F. Moya-Anegon, “Approximate Personal Name- Matching through Finite-State Graphs,” J. Am. Soc. for Information Science and Technology, vol. 58, pp. 1-17, 2007.

[12] M. Bilenko and R. Mooney, “Adaptive Duplicate Detection Using Learnable String Similarity Measures,” Proc. SIGKDD ’03, 2003.

[13] T. Hokama and H. Kitagawa, “Extracting Mnemonic Names of People from the Web,” Proc. Ninth Int’l Conf. Asian Digital Libraries (ICADL ’06), pp. 121-130, 2006.

[14] M. Hearst, “Automatic Acquisition of Hyponyms from Large Text Corpora,” Proc. Int’l Conf. Computational Linguistics (COLING ’92), pp. 539-545, 1992.

[15] M. Berland and E. Charniak, “Finding Parts in Very Large Corpora,” Proc. Ann. Meeting of the Assoc. for Computational Linguistics (ACL ’99), pp. 57-64, 1999.

[16] S. Chakrabarti, Mining the Web: Discovering Knowledge from Hypertext Data. Morgan Kaufmann, 2003.

[17] G. Salton and C. Buckley, “Term-Weighting Approaches in Automatic Text Retrieval,” Information Processing and Management, vol. 24, pp. 513-523, 1988.