CU-COMSEM: Exploring Rich Features for Unsupervised Web Personal Name Disambiguation

Ying Chen
Center for Spoken Language Research
University of Colorado at Boulder
yc@colorado.edu

James Martin
Department of Computer Science
University of Colorado at Boulder
James.Martin@colorado.edu

Abstract

The increasing number of web sources is exacerbating the named-entity ambiguity problem. This paper explores the use of various token-based and phrase-based features in unsupervised clustering of web pages containing personal names. From these experiments, we find that the use of rich features can significantly improve the disambiguation performance for web personal names.

1 Introduction

As the sheer amount of web information expands at an ever more rapid pace, the named-entity ambiguity problem becomes more and more serious in many fields, such as information integration, cross-document co-reference, and question answering. Individuals are so glutted with information that searching for data presents real problems. It is therefore crucial to develop methodologies that can efficiently disambiguate the ambiguous names from any given set of data.

In the paper, we present an approach that combines unsupervised clustering methods with rich feature extractions to automatically cluster returned web pages according to which named entity in reality the ambiguous personal name in a web page refers to. We make two contributions to approaches to web personal name disambiguation. First, we seek to go beyond the kind of bag-of-words features employed in earlier systems (Bagga & Baldwin, 1998; Gooi & Allan, 2004; Pedersen et al., 2005), and attempt to exploit deep semantic features beyond the work of Mann & Yarowsky (2003). Second, we exploit some features that are available only in a web corpus, such as URL information and related web pages.

The paper is organized as follows. Section 2 introduces our rich feature extractions along with their corresponding similarity matrix learning. In Section 3, we analyze the performance of our system. Finally, we draw some conclusions.

2 Methodology

Our approach follows a common architecture for named-entity disambiguation: the detection of ambiguous objects, feature extractions and their corresponding similarity matrix learning, and finally clustering.

Given a webpage, we first run a modified Beautiful Soup[1] (a HTML parser) to extract a clean text document for that webpage. In a clean text document, noisy tokens, such as HTML tags and java codes, are removed as much as possible, and sentence segmentation is partially done by following the indications of some special HTML tags. For example, a sentence should finish when it meets a "<table>" tag. Then each clean document continues to be preprocessed with MXTERMINATOR (a sentence segmenter),[2] the Penn Treebank tokenization,[3] a syntactic phrase chunker (Hacioglu, 2004), and a named-entity detection and co-reference system for the ACE project[4] called EX-

[1] http://www.crummy.com/software/BeautifulSoup
[2] http://www.id.cbs.dk/~dh/corpus/tools/MXTERMINATOR.html
[3] http://www.cis.upenn.edu/~treebank/tokenization.html
[4] http://www.nist.gov/speech/tests/ace

Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007), pages 125–128,
Prague, June 2007.

ERT[5] (Hacioglu et al. 2005; Chen & Hacioglu, 2006).

2.1 The detection of ambiguous objects

For a given ambiguous personal name, for each web page, we try to extract all mentions of the ambiguous personal name, using three possible varieties of the personal name. For example, the three regular expression patterns for "Alexander Markham" are "Alexander Markham," "Markham, Alexander," and "Alexander .\. Markham" (".\." can match a middle name). Web pages without any mention of the ambiguous personal name of interest are discarded and receive no further processing.

Since it is common for a single document to contain one or more mentions of the ambiguous personal name of interest, there is a need to define the object to be disambiguated. Here, we adopt the policy of "one person per document" (all mentions of the ambiguous personal name in one web page are assumed to refer to the same personal entity in reality) as in Bagga & Baldwin (1998), Mann & Yarowsky (2003) and Gooi & Allan (2004). We therefore define an object as a single entity with the ambiguous personal name in a given web page. This definition of the object (document-level object) might be mistaken, because the mentions of the ambiguous personal name in a web page may refer to multiple entities, but we found that this is a rare case (most of those cases occur in genealogy web pages). On the other hand, a document-level object can include much information derived from that web page, so that it can be represented by rich features.

Given this definition of an object, we define a target entity as an entity (outputted from the EXERT system) that includes a mention of the ambiguous personal name. Then, we define a local sentence as a sentence that contains a mention of any target entity.

2.2 Feature extraction and similarity matrix learning

Most of the previous work (Bagga & Baldwin, 1998; Gooi & Allan; 2004; Pedersen et al., 2005) uses token information in the given documents. In this paper, we follow and extend their work especially for a web corpus. On the other hand, compared to a token, a phrase contains more information for named-entity disambiguation. Therefore, we explore some phrase-based information in this paper. Finally, there are two kinds of feature vectors developed in our system: token-based and phrase-based. A token-based feature vector is composed of tokens, and a phrase-based feature vector is composed of phrases.

2.2.1 Token-based features

There is a lot of token information available in a web page: the tokens occurring in that web page, the URL for that web page, and so on. Here, for each web page, we tried to extract tokens according to the following schemes.

Local tokens (Local): the tokens occurring in the local sentences in a given webpage;

Full tokens (Full): the tokens occurring in a given webpage;

URL tokens (URL): the tokens occurring in the URL of a given webpage. URL tokenization works as follows: split a URL at ":" and ".", and then filter out stop words that are very common in URLs, such as "com," "http," and so on;

Title tokens in root page (TTRP): the title tokens occurring in the root page of a given webpage. Here, we define the root page of a given webpage as the page whose URL is the first slash-demarcated element (non-http) of the URL of the given webpage. For example, the root page of "http://www.leeds.ac.uk/calendar/court.htm" is "www.leeds.ac.uk". We do not use all tokens in the root page because there may be a lot of noisy information.

Although Local tokens and Full tokens often provide enough information for name disambiguation, there are some ambiguity cases that can be solved only with the help of information beyond the given web page, such as URL tokens and TTRP tokens. For example, in the web page "Alexander Markham 009," there is not sufficient information to identify the "Alexander Markham." But from its URL tokens ("leeds ac uk calendar court") and the title tokens in its root page ("University of Leeds"), it is easy to infer that this "Alexander Markham" is from the University of Leeds, which can totally solve the name ambiguity.

Because of the noisy information in URL tokens and TTRP tokens, here we combine them with Local tokens, using the following policy: for

[5] http://sds.colorado.edu/EXERT

each URL token and TTRP token, if the token is also one of the Local tokens of other web pages, add this token into the Local token list of the current webpage. We do the same thing with Full tokens.

Except URL tokens, the other three kinds of tokens—Local tokens, Full tokens and TTRP tokens—are outputted from the Penn Treebank tokenization, filtered by a stop-word dictionary, and represented in their morphological root form. But tokens in web pages have special characteristics and need more post-processing. In particular, a token may be an email address or a URL that may contain some useful information. For example, "charlotte@la-par.org" indicates the "Charlotte Bergeron" who works for PAR (the Public Affairs Research Council) in LA (Los Angeles). To capture the fine-grained information in an email address or a URL, we do deep tokenization on these two kinds of tokens. For a URL, we do deep tokenization as URL tokenization; for an email address, we split the email address at "@" and ".", then filter out the stop words as in URL tokenization.

So far, we have developed two token-based feature vectors: a Local token feature vector and a Full token feature vector. Both of them may contain URL and TTRP tokens. Given feature vectors, we need to find a way to learn the similarity matrix. Here, we choose the standard TF-IDF method to calculate the similarity matrix.

2.2.2 Phrase-based features

Since considerable information related to the ambiguous object resides in the noun phrases in a web page, such as the person's job and the person's location, we attempt to capture this noun phrase information. The following section briefly describes how to extract and use the noun phrase information. For more detail, see Chen & Martin (2007).

Contextual base noun phrase feature: With the syntactic phrase chunker, we extract all base noun phrases (non-overlapping syntactic phrases) occurring in the local sentences, which usually include some useful information about the ambiguous object. A base noun phrase of interest serves as an element in the feature vector.

Document named-entity feature: Given the EXERT system, a direct and simple way to use the semantic information is to extract all named entities in a web page. Since a given entity can be represented by many mentions in a document, we choose a single representative mention to represent each entity. The representative mention is selected according to the following ordered preference list: longest NAME mention, longest NOMINAL mention. A representative mention phrase serves as an element in a feature vector.

Given a pair of feature vectors consisting of phrase-based features, we need to choose a similarity scheme to calculate the similarity matrix. Because of the word-space delimiter in English, the feature vector comprises phrases, so that a similarity scheme for phrase-based feature vectors is required. Chen & Martin (2007) introduced one of those similarity schemes, "two-level SoftTFIDF". First, a token-based similarity scheme, the standard SoftTFIDF (Cohen et al., 2003), is used to calculate the similarity between phrases in the pair of feature vectors; in the second phase, the standard SoftTFIDF is reformulated to calculate the similarity for the pair of phrased-based feature vectors.

First, we introduce the standard SoftTFIDF. In a pair of feature vectors S and T, $S = (s_1, \ldots, s_n)$ and $T = (t_1, \ldots, t_m)$. Here, s_i ($i = 1 \ldots n$) and t_j ($j = 1 \ldots m$) are substrings (tokens). Let $CLOSE(\theta; S;T)$ be the set of substrings $w \in S$ such that there is some $v \in T$ satisfying $dist(w; v) > \theta$. The Jaro-Winkler distance function (Winkler, 1999) is $dist(;)$. For $w \in CLOSE(\theta; S;T)$, let $D(w; T) = \max_{v \in T} dist(w;v)$. Then the standard SoftTFIDF is computed as

$$\text{SoftTFIDF}(S,T) = \sum_{w \in CLOSE(\theta;S;T)} V(w, S) \times V(w, T) \times D(w, T)$$

$$V'(w, S) = \log(TF_{w,S} + 1) \times \log(IDF_w)$$

$$V(w, S) = \frac{V'(w, S)}{\sqrt{\sum_{w \in S} V'(w, S)^2}},$$

where $TF_{w,S}$ is the frequency of substrings w in S, and IDF_w is the inverse of the fraction of documents in the corpus that contain w. To compute the similarity for the phrase-based feature vectors, in the second step of "two-level SoftTFIDF," the substring w is a phrase and *dist* is the standard SoftTFIDF.

So far, we have developed several feature models and learned the corresponding similarity ma-

trices, but clustering usually needs only one unique similarity matrix. In the results reported here, we simply combine the similarity matrices, assigning equal weight to each one.

2.3 Clustering

Although clustering is a well-studied area, a remaining research problem is to determine the optimal parameter settings during clustering, such as the number of clusters or the stop-threshold, a problem that is important for real tasks and that is not at all trivial. Because currently we focus only on feature development, we choose agglomerative clustering with a single linkage, and simply use a fixed stop-threshold acquired from the training data.

3 Performance

Our system performs very well for the Semeval Web People corpus, and Table 1 shows the performances. There are two results in Table 1: One is gotten from the evaluation of Semeval Web People Track (SemEval), and the other is evaluated with B-cubed evaluation (Bagga and Baldwin, 1998). Both scores indicate that web personal name disambiguation needs more effort.

	Purity	Inverse Purity	F (α=0.5)	F (α=0.2)
SemEval	0.72	0.88	0.78	0.83
	Precision	Recall	F (α=0.5)	F (α=0.2)
B-cubed	0.61	0.83	0.70	0.77

Table 1 *The performances of the test data*

4 Conclusion

Our experiments in web personal name disambiguation extend token-based information to a web corpus, and also include some noun phrase-based information. From our experiment, we first find that it is not easy to extract a clean text document from a webpage because of much noisy information in it. Second, some common tools need to be adapted to a web corpus, such as sentence segmentation and tokenization. Many NLP tools are developed for a news corpus, whereas a web corpus is noisier and often needs some specific processing. Third, in this paper, we use some URL information and noun phrase information in a rather simple way; more exploration is needed in the future. Besides the rich feature extraction, we also need more work on similarity combination and clustering.

Acknowledgements

Special thanks are extended to Praful Mangalath and Kirill Kireyev.

References

J. Artiles, J. Gonzalo. and S. Sekine. 2007. *The SemEval-2007 WePS Evaluation: Establishing a benchmark for the Web People Search Task*. In Proceedings of Semeval 2007, Association for Computational Linguistics.

A. Bagga and B. Baldwin. 1998. *Entity–based Cross–document Co–referencing Using the Vector Space Model*. In 17th COLING.

Y. Chen and K. Hacioglu. 2006. *Exploration of Coreference Resolution: The ACE Entity Detection and Recognition Task*. In 9th International Conference on TEXT, SPEECH and DIALOGUE.

Y. Chen and J. Martin. 2007. *Towards Robust Unsupervised Personal Name Disambiguation*. EMNLP.

W. Cohen, P. Ravikumar, S. Fienberg. 2003. *A Comparison of String Metrics for Name-Matching Tasks*. In IJCAI-03 II-Web Workshop.

C. H. Gooi and J. Allan. 2004. *Cross-Document Coreference on a Large Scale Corpus*. NAACL

K. Hacioglu, B. Douglas and Y. Chen. 2005. *Detection of Entity Mentions Occurring in English and Chinese Text*. Computational Linguistics.

K. Hacioglu. 2004. *A Lightweight Semantic Chunking Model Based On Tagging*. In HLT/NAACL.

B. Malin. 2005. *Unsupervised Name Disambiguation via Social Network Similarity*. SIAM.

G. Mann and D. Yarowsky. 2003. *Unsupervised Personal Name Disambiguation*. In Proc. of CoNLL-2003, Edmonton, Canada.

T. Pedersen, A. Purandare and A. Kulkarni. 2005. *Name Discrimination by Clustering Similar Contexts*. In Proc. of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics, pages 226-237. Mexico City, Mexico.

W. E. Winkler. 1999. *The state of record linkage and current research problems. Statistics of Income Division*, Internal Revenue Service Publication R99/04.