Extracting Key Phrases to Disambiguate Personal Name Queries in Web Search

Danushka Bollegala **Yutaka Matsuo** * **Mitsuru Ishizuka**
Graduate School of Information Science and Technology
The University of Tokyo
7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan
danushka@mi.ci.i.u-tokyo.ac.jp
y.matsuo@aist.go.jp
ishizuka@i.u-tokyo.ac.jp

Abstract

Assume that you are looking for information about a particular person. A search engine returns many pages for that person's name. Some of these pages may be on other people with the same name. One method to reduce the ambiguity in the query and filter out the irrelevant pages, is by adding a phrase that uniquely identifies the person we are interested in from his/her namesakes. We propose an unsupervised algorithm that extracts such phrases from the Web. We represent each document by a *term-entity* model and cluster the documents using a contextual similarity metric. We evaluate the algorithm on a dataset of ambiguous names. Our method outperforms baselines, achieving over 80% accuracy and significantly reduces the ambiguity in a web search task.

1 Introduction

The Internet has grown into a collection of billions of web pages. Web search engines are important interfaces to this vast information. We send simple text queries to search engines and retrieve web pages. However, due to the ambiguities in the queries, a search engine may return a lot of irrelevant pages. In the case of personal name queries, we may receive web pages for other people with the same name (*namesakes*). For example, if we search *Google* [1] for *Jim Clark*, even among the top 100 results we find at least eight different *Jim Clarks*. The two popular namesakes; *Jim Clark* the Formula one world champion (46 pages), and *Jim Clark* the founder of Netscape (26 pages), cover the majority of the pages. What if we are interested only in the Formula one world champion and want to filter out the pages for the other *Jim Clarks*? One solution is to modify our query by including a phrase such as *Formula one* or *racing driver* with the name, *Jim Clark*.

This paper presents an automatic method to extract such phrases from the Web. We follow a three-stage approach. In the first stage we represent each document containing the ambiguous name by a *term-entity* model, as described in section 5.2. We define a contextual similarity metric based on snippets returned by a search engine, to calculate the similarity between term-entity models. In the second stage, we cluster the documents using the similarity metric. In the final stage, we select key phrases from the clusters that uniquely identify each namesake.

2 Applications

Two tasks that can readily benefit from automatically extracted key phrases to disambiguate personal names are *query suggestion* and *social network extraction*. In query suggestion (Gauch and Smith, 1991), the search engine returns a set of phrases to the user alongside with the search results. The user can then modify the original query using these phrases to narrow down the search. Query suggestion helps the users to easily navigate through the result set. For personal name queries, the key phrases extracted by our algorithm can be used as suggestions to reduce the ambiguity and narrow down the search on a particular namesake.

Social networking services (SNSs) have been given much attention on the Web recently. As a kind of online applications, SNSs can be used

*National Institute of Advanced Industrial Science and Technology

[1] www.google.com

to register and share personal information among friends and communities. There have been recent attempts to extract social networks using the information available on the Web [2](Mika, 2004; Matsuo et al., 2006). In both Matsuo's (2006) and Mika's (2004) algorithms, each person is represented by a node in the social network and the strength of the relationship between two people is represented by the length of the edge between the corresponding two nodes. As a measure of the strength of the relationship between two people *A* and *B*, these algorithms use the number of hits obtained for the query *A AND B*. However, this approach fails when *A* or *B* has namesakes because the number of hits in these cases includes the hits for the namesakes. To overcome this problem, we could include phrases in the query that uniquely identify *A* and *B* from their namesakes.

3 Related Work

Person name disambiguation can be seen as a special case of word sense disambiguation (WSD) (Schutze, 1998; McCarthy et al., 2004) problem which has been studied extensively in Natural Language Understanding. However, there are several fundamental differences between WSD and person name disambiguation. WSD typically concentrates on disambiguating between 2-4 possible meanings of the word, all of which are a priori known. However, in person name disambiguation in Web, the number of different namesakes can be much larger and unknown. From a resource point of view, WSD utilizes sense tagged dictionaries such as WordNet, whereas no dictionary can provide information regarding different namesakes for a particular name.

The problem of person name disambiguation has been addressed in the domain of research paper citations (Han et al., 2005), with various supervised methods proposed for its solution. However, citations have a fixed format compared to free text on the Web. Fields such as co-authors, title, journal name, conference name, year of publication can be easily extracted from a citation and provide vital information to the disambiguation process.

Research on multi-document person name resolution (Bagga and Baldwin, 1998; Mann and Yarowsky, 2003; Fleischman and Hovy, 2004) focuses on the related problem of determining if two instances with the same name and from different documents refer to the same individual. Bagga and Baldwin (1998) first perform within-document coreference resolution to form coreference chains for each entity in each document. They then use the text surrounding each reference chain to create summaries about each entity in each document. These summaries are then converted to a bag of words feature vector and are clustered using standard vector space model often employed in IR. The use of simplistic bag of words clustering is an inherently limiting aspect of their methodology. On the other hand, Mann and Yarowsky (2003) proposes a richer document representation involving automatically extracted features. However, their clustering technique can be basically used only for separating two people with the same name. Fleischman and Hovy (2004) constructs a maximum entropy classifier to learn distances between documents that are then clustered. Their method requires a large training set.

Pedersen et al. (2005) propose an unsupervised approach to resolve name ambiguity by representing the context of an ambiguous name using second order context vectors derived using singular value decomposition (SVD) on a co-occurrence matrix. They agglomeratively cluster the vectors using cosine similarity. They evaluate their method only on a conflated dataset of pseudonames, which begs the question of how well such a technique would fair on a more real-world challenge. Li et al. (2005) propose two approaches to disambiguate entities in a set of documents: a supervisedly trained pairwise classifier and an unsupervised generative model. However, they do not evaluate the effectiveness of their method in Web search.

Bekkerman and McCallum (2005) present two unsupervised methods for finding web pages referring to a particular person: one based on link structure and another using Agglomerative/Conglomerative Double Clustering (A/CDC). Their scenario focuses on simultaneously disambiguating an existing social network of people, who are closely related. Therefore, their method cannot be applied to disambiguate an individual whose social network (for example, friends, colleagues) is not known. Guha and Grag (2004) present a re-ranking algorithm to disambiguate people. The algorithm requires a user to select one of the returned pages as a starting point. Then,

[2]http://flink.sematicweb.org/. The system won the 1st place at the Semantic Web Challenge in ISWC2004.

Table 1: Data set for experiments

Collection	No of namesakes
person-X	4
Michael Jackson	3
Jim Clark	8
William Cohen	10

through comparing the person descriptions, the algorithm re-ranks the entire search results in such a way that pages referring to the same person described in the user-selected page are ranked higher. A user needs to browse the documents in order to find which matches the user's intended referent, which puts an extra burden on the user.

None of the above mentioned works attempt to extract key phrases to disambiguate person name queries, a contrasting feature in our work.

4 Data Set

We select three ambiguous names (*Micheal Jackson*, *William Cohen* and *Jim Clark*) that appear in previous work in name resolution. For each name we query *Google* with the name and download top 100 pages. We manually classify each page according to the namesakes discussed in the page. We ignore pages which we could not decide the namesake from the content. We also remove pages with images that do not contain any text. No pages were found where more than one namesakes of a name appear. For automated pseudo-name evaluation purposes, we select four names (*Bill Clinton*, *Bill Gates*, *Tom Cruise* and *Tiger Woods*) for conflation, who we presumed had one vastly predominant sense. We download 100 pages from Google for each person. We replace the name of the person by "person-X" in the collection, thereby introducing ambiguity. The structure of our dataset is shown in Table 1.

5 Method

5.1 Problem Statement

Given a collection of documents relevant to an ambiguous name, we assume that each document in the collection contains exactly one namesake of the ambiguous name. This is a fair assumption considering the fact that although namesakes share a common name, they specializes in different fields and have different Web appearances. Moreover, the one-to-one association between documents and people formed by this assumption, let us model the person name disambiguation problem as a one of hard-clustering of documents.

The outline of our method is as following; Given a set of documents representing a group of people with the same name, we represent each document in the collection using a *Term-Entity* model (section 5.2). We define a contextual similarity metric (section 5.4) and then cluster (section 5.5) the term-entity models using the contextual similarity between them. Each cluster is considered to be representing a different namesake. Finally, key phrases that uniquely identify each namesake are selected from the clusters. We perform experiments at each step of our method to evaluate its performance.

5.2 Term-Entity Model

The first step toward disambiguating a personal name is to identify the discriminating features of one person from another. In this paper we propose *Term-Entity models* to represent a person in a document.

Definition. *A term-entity model $T(A)$, representing a person A in a document D, is a boolean expression of n literals $a_1, a_2, \ldots, a_n$. Here, a boolean literal a_i is a multi-word term or a named entity extracted from the document D.*

For simplicity, we only consider boolean expressions that combine the literals through AND operator.

The reasons for using terms as well as named entities in our model are two fold. Firstly, there are multi-word phrases such as *secretary of state*, *racing car driver* which enable us to describe a person uniquely but not recognized by named entity taggers. Secondly, automatic term extraction (Frantzi and Ananiadou, 1999) can be done using statistical methods and does not require extensive linguistic resources such as named entity dictionaries, which may not be available for some domains.

5.3 Creating Term-Entity Models

We extract terms and named entities from each document to build the term-entity model for that document. For automatic multi-word term extraction, we use the *C-value* metric proposed by Frantzi et al. (1999). Firstly, the text from which we need to extract terms is tagged using a part of speech tagger. Then a linguistic filter and a stop words list constrain the word sequences that

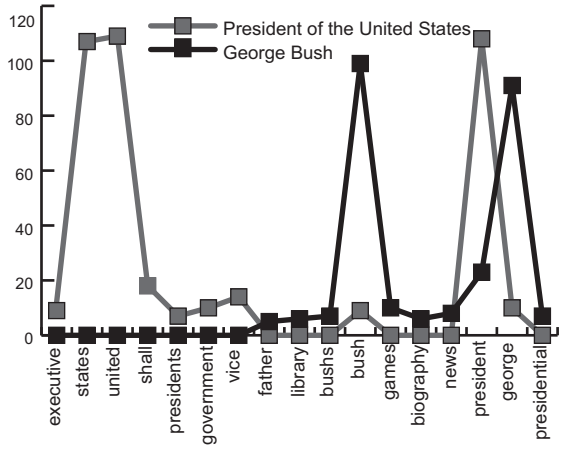


Figure 1: Distribution of words in snippets for "George Bush" and "President of the United States"

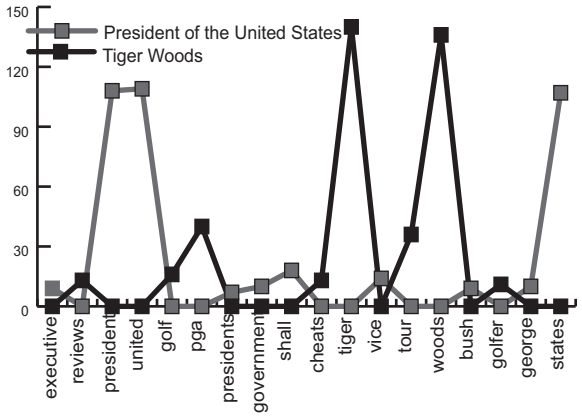


Figure 2: Distribution of words in snippets for "Tiger Woods" and "President of the United States"

are allowed as genuine multi-word terms. The linguistic filter contains a predefined set of patterns of nouns, adjectives and prepositions that are likely to be terms. The sequences of words that remain after this initial filtering process (candidate terms) are evaluated for their *termhood* (likeliness of a candidate to be a term) using C-value. C-value is built using statistical characteristics of the candidate string, such as, total frequency of occurrence of the candidate string in the document, the frequency of the candidate string as part of other longer candidate strings, the number of these longer candidate terms and the length of candidate string (in number of words). We select the candidates with higher C-values as terms (see (Frantzi and Ananiadou, 1999) for more details on C-value based term extraction).

To extract entities for the term-entity model, the documents were annotated by a named entity tagger [3]. We select personal names, organization names and location names to be included in the term-entity model.

5.4 Contextual Similarity

We need to calculate the similarity between term-entity models derived from different documents, in order to decide whether they belong to the same namesake or not. WordNet [4] based similarity metrics have been widely used to compute the semantic similarity between words in sense disambiguation tasks (Banerjee and Pedersen, 2002; McCarthy et al., 2004). However, most of the terms and entities in our term-entity models are proper names or multi-word expressions which are not listed in WordNet.

Sahami et al. (2005) proposed the use of snippets returned by a Web search engine to calculate the semantic similarity between words. A snippet is a brief text extracted from a document around the query term. Many search engines provide snippets alongside with the link to the original document. Since snippets capture the immediate surrounding of the query term in the document, we can consider a snippet as the context of a query term. Using snippets is also efficient because we do not need to download the source documents. To calculate the contextual similarity between two terms (or entities), we first collect snippets for each term (or entity) and pool the snippets into a combined "bag of words". Each collection of snippets is represented by a word vector, weighted by the normalized frequency (i.e., frequency of a word in the collection is divided by the total number of words in the collection). Then, the contextual similarity between two phrases is defined as the inner product of their snippet-word vectors.

Figures 1 and 2 show the distribution of most frequent words in snippets for the queries "George Bush", "Tiger Woods" and "President of the United States". In Figure 1 we observe the words "george" and "bush" appear in snippets for the query "President of the United States", whereas in Figure 2 none of the high frequent words appears in snippets for both queries. Contextual

[3] The named entity tagger was developed by the Cognitive Computation Group at UIUC. http://L2R.cs.uiuc.edu/ cogcomp/eoh/ne.html

[4] http://wordnet.princeton.edu/perl/webwn

similarity calculated as the inner product between word vectors is 0.2014 for "George Bush" and "President of the United States", whereas the same is 0.0691 for "Tiger Woods" and "President of the United States". We define the similarity $\mathrm{sim}(T(A), T(B))$, between two term-entity models $T(A) = \{a_1, \ldots, a_n\}$ and $T(B) = \{b_1, \ldots, b_m\}$ of documents A and B as follows,

$$\mathrm{sim}(T(A), T(B)) = \frac{1}{n} \sum_{i=1}^{n} \max_{1 \leq j \leq m} |a_i| \cdot |b_j|. \quad (1)$$

Here, $|a_i|$ represents the vector that contains the frequency of words that appear in the snippets for term/entity a_i. Contextual similarity between terms/entities a_i and b_j, is defined as the inner product $|a_i| \cdot |b_j|$. Without a loss of generality we assume $n \leq m$ in formula 1.

5.5 Clustering

We use Group-average agglomerative clustering (GAAC) (Cutting et al., 1992), a hybrid of single-link and complete-link clustering, to group the documents that belong to a particular namesake. Initially, we assign a separate cluster for each of the documents in the collection. Then, GAAC in each iteration executes the merger that gives rise to the cluster Γ with the largest average correlation $C(\Gamma)$ where,

$$C(\Gamma) = \frac{1}{2} \frac{1}{|\Gamma|(|\Gamma| - 1)} \sum_{u \in \Gamma} \sum_{v \in \Gamma} \mathrm{sim}(T(u), T(v)) \quad (2)$$

Here, $|\Gamma|$ denotes the number of documents in the merged cluster Γ; u and v are two documents in Γ and $\mathrm{sim}(T(u), T(v))$ is given by equation 1. Determining the total number of clusters is an important issue that directly affects the accuracy of disambiguation. We will discuss an automatic method to determine the number of clusters in section 6.3.

5.6 Key phrases Selection

GAAC process yields a set of clusters representing each of the different namesakes of the ambiguous name. To select key phrases that uniquely identify each namesake, we first pool all the terms and entities in all term-entity models in each cluster. For each cluster we select the most discriminative terms/entities as the key phrases that uniquely identify the namesake represented by that cluster from the other namesakes. We achieve this in two steps. In the first step, we reduce the number of terms/entities in each cluster by removing terms/entities that also appear in other clusters. In the second step, we select the terms/entities in each cluster according to their relevance to the ambiguous name. We compute the contextual similarity between the ambiguous name and each term/entity and select the top ranking terms/entities from each cluster.

6 Experiments and Results

6.1 Evaluating Contextual Similarity

In section 5.4, we defined the similarity between documents (i.e., term-entity models created from the documents) using a web snippets based contextual similarity (Formula 1). However, how well such a metric represents the similarity between documents, remains unknown. Therefore, to evaluate the contextual similarity among documents, we group the documents in "person-X" dataset into four classes (each class representing a different person) and use Formula 1 to compute within-class and cross-class similarity histograms, as illustrated in Figure 3.

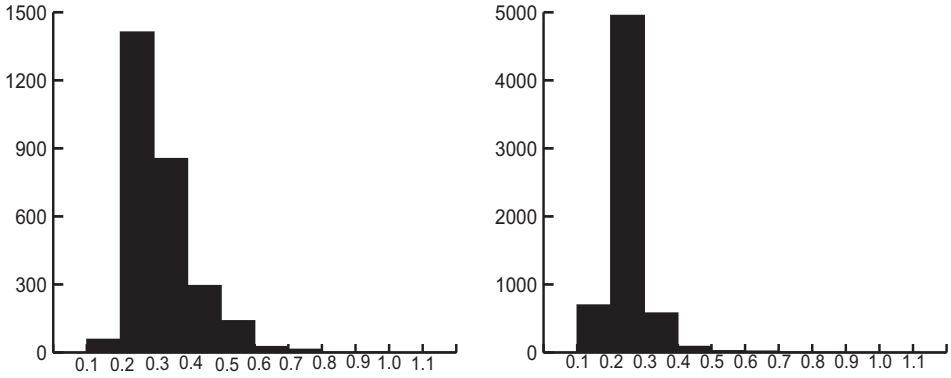
Ideally, within-class similarity distribution should have a peak around 1 and cross-class similarity distribution around 0, whereas both histograms in Figure 3(a) and 3(b) have their peaks around 0.2. However, within-class similarity distribution is heavily biased toward to the right of this peak and cross-class similarity distribution to the left. Moreover, there are no document pairs with more than 0.5 cross-class similarity. The experimental results guarantees the validity of the contextual similarity metric.

6.2 Evaluation Metric

We evaluate experimental results based on the confusion matrix, where $A[i.j]$ represents the number of documents of "person i" predicted as "person j" in matrix A. $A[i, i]$ represents the number of correctly predicted documents for "person i". We define the disambiguation accuracy as the sum of diagonal elements divided by the sum of all elements in the matrix.

6.3 Cluster Quality

Each cluster formed by the GAAC process is supposed to be representing a different namesake. Ideally, the number of clusters formed should be equal to the number of different namesakes for



(a) Within-class similarity distribution in "person-X" dataset

(b) Cross-class similarity distribution in "person-X" dataset

Figure 3: The histogram of within-class and cross-class similarity distributions in "person-X" dataset. X axis represents the similarity value. Y axis represents the number of document pairs from the same class (within-class) or from different classes (cross-class) that have the corresponding similarity value.

the ambiguous name. However, in reality it is impossible to exactly know the number of namesakes that appear on the Web for a particular name. Moreover, the distribution of pages among namesakes is not even. For example, in the "Jim Clark" dataset 78% of documents belong to the two famous namesakes (*CEO Nestscape* and *Formula one world champion*). The rest of the documents are distributed among the other six namesakes. If these outliers get attached to the otherwise pure clusters, both disambiguation accuracy and key phrase selection deteriorate. Therefore, we monitor the *quality* of clustering and terminate further agglomeration when the cluster quality drops below a pre-set threshold. Numerous metrics have been proposed for evaluating quality of clustering (Kannan et al., 2000). We use normalized cuts (Shi and Malik, 2000) as a measure of cluster-quality.

Let, V denote the set of documents for a name. Consider, $A \subseteq V$ to be a cluster of documents taken from V. For two documents x,y in V, $\mathrm{sim}(x, y)$ represents the contextual similarity between the documents (Formula 1). Then, the normalized cut $N_{cut}(A)$ of cluster A is defined as,

$$N_{cut}(A) = \frac{\sum_{x \in A\, y \in (V-A)} \mathrm{sim}(x, y)}{\sum_{x \in A\, y \in V} \mathrm{sim}(x, y)}. \quad (3)$$

For a set, $\{A_1, \ldots, A_n\}$ of non-overlapping n clusters A_i, we define the *quality* of clustering, $\mathrm{Quality}(\{A_1, \ldots, A_n\})$, as follows,

$$\mathrm{Quality}(\{A_1, \ldots, A_n\}) = \frac{1}{n} \sum_{i=1}^{n} N_{cut}(A_i). \quad (4)$$

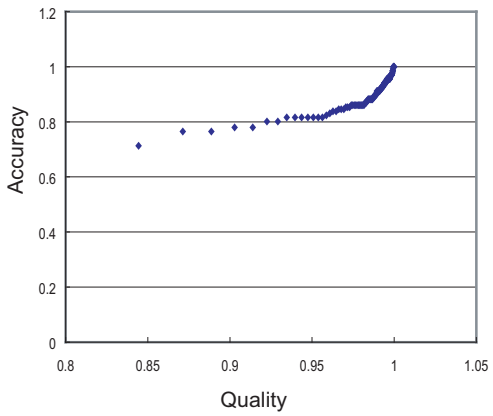


Figure 4: Accuracy Vs Cluster Quality for person-X data set.

To explore the faithfulness of cluster quality in approximating accuracy, we compare accuracy (calculated using human-annotated data) and cluster quality (automatically calculated using Formula 4) for person-X data set. Figure 4 shows cluster quality in x-axis and accuracy in y-axis. We observe a high correlation (Pearson coefficient of 0.865) between these two measures, which enables us to guide the clustering process through cluster quality.

When cluster quality drops below a pre-defined

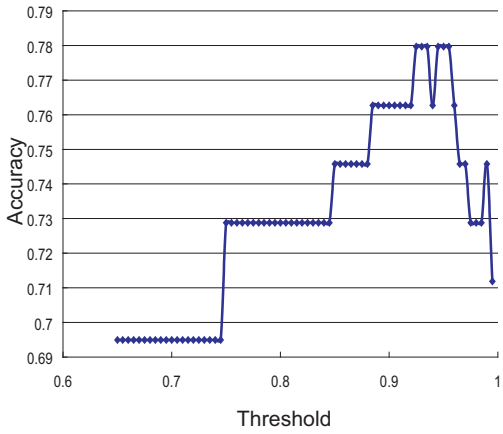


Figure 5: Accuracy Vs Threshold value for person-X data set.

threshold, we terminate further clustering. We assign the remaining documents to the already formed clusters based on the correlation (Formula 2) between the document and the cluster. To determine the threshold of cluster quality, we use person-X collection as training data. Figure 5 illustrates the variation of accuracy with threshold. We select threshold at 0.935 where accuracy maximizes in Figure 5. Threshold was fixed at 0.935 for the rest of the experiments.

6.4 Disambiguation Accuracy

Table 2 summarizes the experimental results. The baseline, majority sense , assigns all the documents in a collection to the person that have most documents in the collection. Proposed method outperforms the baseline in all data sets. Moreover, the accuracy values for the proposed method in Table 2 are statistically significant (t-test: P(T$\leq$t)=0.0087, $\alpha = 0.05$) compared to the baseline. To identify each cluster with a namesake, we chose the person that has most number of documents in the cluster. "Found" column shows the number of correctly identified namesakes as a fraction of total namesakes. Although the proposed method correctly identifies the popular namesakes, it fails to identify the namesakes who have just one or two documents in the collection.

6.5 Web Search Task

Key phrases extracted by the proposed method are listed in Figure 6 (Due to space limitations, we show only the top ranking key phrases for two collections). To evaluate key phrases in disambiguating namesakes, we set up a web search experiment as follows. We search for the ambiguous name and the key phrase (for example, *"Jim Clark"* AND *"racing driver"*) and classify the top 100 results according to their relevance to each namesake. Results of our experiment on *Jim Clark* dataset for the top ranking key phrases are shown in Table 3.

In Table 3 we classified Google search results into three categories. "person-1" is the formula one racing world champion, "person -2" is the founder of Netscape and "other" category contains rest of the pages that we could not classify to previous two groups [5]. We first searched Google without adding any key phrases to the name. Including terms *racing diver*, *rally* and *scotsman*,

Table 2: Disambiguation accuracy for each collection.

Collection	Majority Sense	Proposed Method	Found Correct
person-X	0.3676	0.7794	4/4
Michael Jackson	0.6470	0.9706	2/3
Jim Clark	0.4407	0.7627	3/8
William Cohen	0.7614	0.8068	3/10

Michael Jackson

CLUSTER #1	CLUSTER #2
fan club	beer hunter
trial	ultimate beer FAQ
world network	christmas beer
superstar	great beer
new charity song	pilsener beer
neverland ranch	barvaria

Jim Clark

CLUSTER #1	CLUSTER #2
racing driver	entrepreneur
rally	story
scotsman	silicon valley
driving genius	CEO
scottish automobile racer	silicon graphics
british rally news	SGI/ Netscape

Figure 6: Top ranking key phrases in clusters for *Michael Jackson* and *Jim Clark* datasets.

Table 3: Effectiveness of key phrases in disambiguating namesakes.

Phrase	person-1	person-2	others	Hits
NONE	41	26	33	1,080,000
racing driver	81	1	18	22,500
rally	42	0	58	82,200
scotsman	67	0	33	16,500
entrepreneur	1	74	25	28,000
story	17	53	30	186,000
silicon valley	0	81	19	46,800

[5] some of these pages were on other namesakes and some were not sufficiently detailed to properly classify

which were the top ranking terms for *Jim Clark* the formula one champion, yields no results for the other popular namesake. Likewise, the key words *entrepreneur* and *silicon valley* yield results fort he founder of Netscape. However, the key word *story* appears for both namesakes. A close investigation revealed that, the keyword *story* is extracted from the title of the book "The New New Thing: A Silicon Valley Story", a book on the founder of Netscape.

7 Conclusion

We proposed and evaluated a key phrase extraction algorithm to disambiguate people with the same name on the Web. We represented each document with a term-entity model and used a contextual similarity metric to cluster the documents. We also proposed a novel approach to determine the number of namesakes. Our experiments with pseudo and naturally ambiguous names show a statistically significant improvement over the baseline method. We evaluated the key phrases extracted by the algorithm in a web search task. The web search task reveals that including the key phrases in the query considerably reduces ambiguity. In future, we plan to extend the proposed method to disambiguate other types of entities such as location names, product names and organization names.

References

A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of COLING*, pages 79–85.

Satanjeev Banerjee and Ted Pedersen. 2002. An adapted lesk algorithm for word sense disambiguation using word net. In *Proceedings of the third international conference on computational linguistics and intelligent text processing*, pages 136–145.

Ron Bekkerman and Andrew McCallum. 2005. Disambiguating web appearances of people in a social network. In *Proceedings of the 14th international conference on World Wide Web*, pages 463–470.

Douglass R. Cutting, Jan O. Pedersen, David Karger, and John W. Tukey. 1992. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings SIGIR '92*, pages 318–329.

M.B. Fleischman and E. Hovy. 2004. Multi-document person name resolution. In *Proceedings of 42nd Annual Meeting of the Association for Computational Linguistics (ACL), Reference Resolution Workshop*.

K.T. Frantzi and S. Ananiadou. 1999. The c-value/nc-value domain independent method for multi-word term extraction. *Journal of Natural Language Processing*, 6(3):145–179.

S. Gauch and J. B. Smith. 1991. Search improvement via automatic query reformulation. *ACM Trans. on Information Systems*, 9(3):249–280.

R. Guha and A. Garg. 2004. Disambiguating people in search. In *Stanford University*.

Hui Han, Hongyuan Zha, and C. Lee Giles. 2005. Name disambiguation in author citations using a k-way spectral clustering method. In *Proceedings of the International Conference on Digital Libraries*.

Ravi Kannan, Santosh Vempala, and Adrian Vetta. 2000. On clusterings: Good, bad, and spectral. In *Proceedings of the 41st Annual Symposium on the Foundation of Computer Science*, pages 367–380.

Xin Li, Paul Morie, and Dan Roth. 2005. Semantic integration in text, from ambiguous names to identifiable entities. *AI Magazine, American Association for Artificial Intelligence*, Spring:45–58.

Gideon S. Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Proceedings of CoNLL-2003*, pages 33–40.

Y. Matsuo, J. Mori, and M. Hamasaki. 2006. Polyphonet: An advanced social network extraction system. In *to appear in World Wide Web Conference (WWW)*.

D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, pages 279–286.

P. Mika. 2004. Bootstrapping the foaf-web: and experiment in social networking network minning. In *Proceedings of 1st Workshop on Friend of a Friend, Social Networking and the Semantic Web*.

Ted Pedersen, Amruta Purandare, and Anagha Kulkarni. 2005. Name discrimination by clustering similar contexts. In *Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics*.

Mehran Sahami and Tim Heilman. 2005. A web-based kernel function for matching short text snippets. In *International Workshop located at the 22nd International Conference on Machine Learning (ICML 2005)*.

Hinrich Schutze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.