Web Person Name Disambiguation by Relevance Weighting of Extended Feature Sets

Chong Long and Lei Shi

Yahoo! Global R&D, Beijing {chongl,lshi}@yahoo-inc.com

Abstract. This paper describes our approach to the Person Name Disambiguation clustering task in the Third Web People Search Evaluation Campaign(WePS-3). The method focuses on two aspects: the extended feature sets, and feature relevance weighting. Bag-of-words and named entities are most commonly used features in many existing web entity disambiguation algorithms and we further extend this basic feature set with Wikipedia concepts. Then two feature weighting models are employed. One is the feature relevance to the target person name(or "query name"), and the other is the feature relevance to the text content. Similarity score is calculated according to the feature weights for clustering documents of the same person. Experiments show that the system based on our approach has generated the best results among all the WePS-3's submissions.

1 Introduction

Person name disambiguation has long been an important problem in natural language processing and text mining. Due to prevalent occurrences on the web that identical person names (or surface names) on different web pages refer to distinct people, being able to resolve the referees of person names on web content is essential for many applications. For instance,

(1) In web search, 15-21% of the queries contain person names (11-17% of the queries are composed of a person name in web search, with aditional terms and 4% are identified simply as person names) [14]. If we are able to retrieve documents that match the user's intended person instead of the surface name, the relevance of search results for people related queries can be substantially improved. (2) Many online social network applications rely on person name as one of the major identities of their users. Resolution of person name ambiguity is hence crucial for many online SNS services. (3) Along with ambiguity of word sense, entity name ambiguity has been a major impediment for many natural language processing tasks, such as text classification, clustering etc.

WePS(http://nlp.uned.es/weps/) is a public evaluation campaign for web entity disambiguation, providing annotated datasets for training and testing [1]. In 2010, we Yahoo! Software R&D Beijing participated the Person Name Disambiguation Task in the third workshop, namely WePS-3. In this task, 300 person names (or query names) are provided along with the top 200 documents retrieved from the search engine for each of the person name. The target is to cluster documents based on the identity of the person, such that documents with names referring the same person are converged into the same cluster.

Our method for the task focuses on two aspects: the feature set and feature weighting. Bag-of-words and named entities are most commonly used features in many existing web entity disambiguation algorithms. Although they have been reported as effective [6], we further extend the basic feature set with Wikipedia concepts. Wikipedia offers a large repository of a wide range of concepts. Compared with conventional named entities consisting of person, location and organization, Wikipedia concepts have many advantages such as well-organized, clean and accurate. For instance, "Support Vector Machine" is better treated as a single coherent conceptual unit rather than three individual words as it is an entry in Wikipedia. Concepts like "Support Vector Machine" are also unable to be recognized by current NER tools because they do not belong to person, location or organization. Since Wikipedia entries are edited by human, they are very accurate compared with entities automatically recognized by NER tools. Previous attempts to leverage Wikipedia for entity disambiguation concentrated on using Wikipedia entries as referees for resolution instead of features. They tried to map a surface name in the text to a Wikipedia entry. However, due to limited coverage of Wikipedia on people, majority of the person names are actually out of Wikipedia except some famous people and therefore this method does not apply to most of the people on the web.

To assign weights to the features that indicate their contribution in resolving the person name's identity, we employed two weighting models. Most of the existing methods use TFIDF as feature weights. Though simple, TFIDF may not well represent the feature's relevance to the query name as well as the content of the text. Some researchers use information extraction method to extract all the related entities. For example, if the sentence "George Bush is the former president of the U.S." fits a pattern, "the former president" will be extracted as the profession of George Bush. But, pattern-based methods normally lead to low recall due to its difficulty to enumerate all the highly accurate patterns between elements. Our method views a feature's contribution to person name disambiguation from two different perspectives. First, the feature should be relevant or related to the query name; second, the feature should represent the content of the text. And accordingly, we employ two weighting models to measure feature relevance in these two regards.

In this paper, we first introduce the related works in Section 2; then we describe complementing the conventional bag-of-words or named entity based features with Wikipedia concepts in Section 3.1. And in Section 3.2 two feature weighting models are introduced. One is the model to measure feature relevance to the query name and the other is the relevance to the text content. Section 3.3 and Section 3.4 present our method to calculate similarity measure and our clustering algorithm, respectively. The experiment results on the WePS datasets are shown in Section 4.

2 Related Work

Web person name disambiguation is also viewed as cross-document co-reference problem in the many previous work. Bagga et al. [12] employed co-occuring word vectors to calculate similarity between entity names. Niu et al. [11] extended Bagga's method through information extraction. Mann and Yarowsky [10] proposed the clus-

2

tering method based on extracted biographic data. However, Niu and Mann's methods were only evaluated on manually generated test data and mainly focus on person name disambiguation. Wan et al. [15] took the assumption that a query entity of a person usually omits the middle name and implemented a person name disambiguation system called "WebHawk". Our approach will be able to deal with more general situations.

Bekkerman and McCallum [3] focused on social network to find documents that refer to a particular person through two methods: one is based on a link structure and the other used agglomerative/conglomerate double clustering. Bunescu and Pasca [4] and Cucerzan [5] used Wikipedia knowledge to disambiguate named entities. However, different from our approach, they try to "map" the surface names in the text to a Wikipedia entry. Due to the limitation of the coverage of the wikipedia entries of people, this method cannot be applied to resolve the majority of the people who are not famous enough to be included in Wikipedia.

Recently Yoshida et al. [16] proposed a two-stage clustering algorithm and further used the bootstrapping algorithm in the second stage [17]. Their method relies heavily on named entity extraction. In Section 4 we will show that our approach that incorporates Wikipedia concepts outperforms those based on entities identified by conventional named entity recognition modules.

3 Methodology

In this section we present our proposed web person name disambiguation approach, which consists of four main steps. The overview of our approach will be provided first, followed by detailed steps.

1. First, Wikipedia concepts are extracted as features of a web page, together with other conventional features such as bag-of-words and named entities. The web page is converted into a feature vector based on the three types of features extracted from the text.

2. Then the weight of each feature in the feature vector is estimated by two weighting models: one is the feature's relevance to the query name, and the other is the relevance to the text content. Each feature in a vector is measured by its TFIDF score and weights under two models.

3. After that the similarity score between two different pages containing the same query name is calculated through their feature vectors based on two similarity measures: cosine similarity and overlap similarity.

4. Finally, web pages referred to the same entity are clustered according to the pairwise similarity score calculated in the previous step.

3.1 Wikipedia Concept Extraction

As mentioned in Section 2, much of the existing work takes named entities as important features. We, in addition, include Wikipedia concepts(or called "Wikipedia elements") extracted from the text in our feature set. We first extract all the manually edited entries from Wikipedia and build a Wikipedia concept dictionary. Given a web page (with html tags removed), the FSA (Finite State Automata) is used to extract string sequences in the

text that match the Wikipedia concepts in the dictionary. In order to avoid overlapping, we use the maximum matching. For instance, both "People's Republic of China" and "China" are Wikipedia concepts. Since "People's Republic of China" contains the string "China" in it, only the maximum match "People's Republic of China" is extracted as a Wikipedia concept feature. These features together with the bag-of-words and named entities of person, location and organization names recognized with the Stanford NER tool (http://nlp.stanford.edu/ner/index.shtml) form a feature vector that represents the content of the web page.

Therefore, our extended feature set has three types of features in all: Wikipedia concepts, bag-of-words and named entities.

Compared with bag-of-words and named entities, using Wikipedia concepts offers the following merits:

1. Wikipedia is a large, well-organized dictionary for named entities. For example, "Support Vector Machines" is treated as three different words under the bag-of-words model. However, with Wikipedia, this term is rather recognized as a single concept since Wikipedia has a manually edited entry for it.

2. Wikipedia's redirect pages can help find other alternative names for an entity [4]. For example, the redirect pages of "United States" correspond to acronyms (U.S.A., U.S., USA, US), Spanish translations (Los Estados Unidos, Estados Unidos), misspellings (Untied States) or synonyms (Yankee land).

3. Wikipedia's disambiguation pages can guide the system to disambiguate a number of entities. For example, the disambiguation page for the name "Michael Jordan" lists 8 associated entities(people). If there is a name "Michael Jordan" in a web page and it is closely related to one of the 8 people, it can help the system to make a decision.

Our experiments on the WePS dataset shows in Section 4 that our system with Wikipedia features outperforms the ones with only bag-of-words and named entity features.

3.2 Feature Weighting Model

After the web page is converted into a feature vector, every feature in the vector is assigned a weight measuring its importance in recognizing the identity of the query name. Each word, named entity and Wikipedia concept, is called a "unit", denoted as u in this paper. These units vary from each other according to their corresponding feature weights. At the beginning, each unit is assigned a TFIDF score

$$TFIDF(u) = tf(u) \bullet -\log df(u) \tag{1}$$

where tf(u) is u's term frequency on the web page, and df(u) is u's document frequency on a large corpus. We use the Yahoo! search engine to collect the statistics of df(u). Then we propose two feature weighting models: the query relevance model and the content relevance model, to assign each unit a proper weight.

Query Relevance Weighting Model Query relevance weighting is to measure how relevant a feature is to the query name. Intuitively, relevant concepts of the query name can better represent its identity. In our method, we base our weighting model on the

assumption that words or concepts that appear close to the query name in the text are more relevant than distant ones. The distance d(u) is measured by the minimum number of sentences between those contain query q and those contain u. d(u) = 0 if u and qco-exist in the same sentence. All 'u's with $0 \le d(u) \le d_{max}$ are considered. We get $d_{max} = 11$ from the training sets. Three polynomial functions are used: $f_1(u)$, $f_2(u)$ and $f_3(u)$. If $d(u) > d_{max}$, $f_1(u) = f_2(u) = f_3(u) = 0$; if $0 \le d(u) \le d_{max}$, they can be computed as Equation 2 to Equation 4.

$$f_1(u) = 1 - \frac{d(u)}{d_{max}} \tag{2}$$

$$f_2(u) = 1 - \left(\frac{d(u)}{d_{max}}\right)^2$$
(3)

$$f_3(u) = (1 - \frac{d(u)}{d_{max}})^2$$
(4)

Here we give an example. The following passage comes from a Wikipedia's article about Michael Jordan (http://en.wikipedia.org/wiki/Michael_Jordan). This passage has the following ten sentences, which are numbered from one to ten.

1. In the 1990 - 91 season, Jordan won his second MVP award after averaging 31.5 ppg on 53.9% shooting, 6.0 rpg, and 5.5 apg for the regular season.

2. The Bulls finished in first place in their division for the first time in 16 years and set a franchise record with 61 wins in the regular season.

3. With Scottie Pippen developing into an All-Star, the Bulls elevated their play.

4. The Bulls defeated the New York Knicks and the Philadelphia 76ers in the opening two rounds of the playoffs.

5. They advanced to the Eastern Conference Finals where their rival, the Detroit Pistons, awaited them.

6. However, this time the Bulls beat the Pistons in a surprising sweep.

7. In an unusual ending to the fourth and final game, Isiah Thomas led his team off the court before the final minute had concluded.

8. Most of the Pistons went directly to their locker room instead of shaking hands with the Bulls.

9. The Bulls compiled an outstanding 15 - 2 record during the playoffs, and advanced to the NBA Finals for the first time in franchise history, where they beat the Los Angeles Lakers four games to one.

10. Perhaps the best known moment of the series came in Game 2 when, attempting a dunk, Jordan avoided a potential Sam Perkins block by switching the ball from his right hand to his left in mid-air to lay the shot in.

The query name is "Michael Jordan", or "Jordan". Each sentence's distance weight is shown in Table 3.2. Two sentences in the passage above contains the query name: No.1 and No.10, therefore, their units' d(u) is 0. The units in the 2nd and the 9th sentences get d(u) = 1 because sentences No.1 and No.10 are close to them, respectively. With this method we can get the other sentences' distances to the query name. The distance weights under three weighting functions are listed in the last three rows, respectively.

Table 1. Distance Weighting for the Query Name "Michael Jordan"

| Sentence # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|------------|------|------|------|------|------|------|------|------|------|------|
| d(u) | 0 | 1 | 2 | 3 | 4 | 4 | 3 | 2 | 1 | 0 |
| $f_1(u)$ | 1.00 | 0.90 | 0.80 | 0.70 | 0.60 | 0.60 | 0.70 | 0.80 | 0.90 | 1.00 |
| $f_2(u)$ | 1.00 | 0.99 | 0.96 | 0.91 | 0.84 | 0.84 | 0.91 | 0.96 | 0.99 | 1.00 |
| $f_3(u)$ | 1.00 | 0.81 | 0.64 | 0.49 | 0.36 | 0.36 | 0.49 | 0.64 | 0.81 | 1.00 |

A unit u's score under query relevance weighting model $v_p(u)$ is its TFIDF score TFIDF(u) multiplied by a distance weight function f(u). f(u) can either be $f_1(u)$, $f_2(u)$ or $f_3(u)$.

If u appears for more than one times within a document, all us' weights in different positions are accumulated together.

$$v_p(u) = TFIDF(u) \cdot \sum_{u \in d} f(u)$$
(5)

where d is a web page.

Content Relevance Weighting Model The content relevance weighting model measures the feature's relevance to the content of the web page. It originates from the observation that query names in the web pages of similar content tend to refer to the same person. Specifically:

1. If the query name is the main topic of the page (say, the home page of a person), all units in the web page are related to the query. Then their relevance to the page content is similar to their relevance to the query name.

2. If the query name is not the main topic of the content, for example, a query name "David Beckham" mentioned in some news article talking about soccer games, though the query names like "David Beckham" are not the main topic of the articles, they tend to refer to the same person in the text with words like "soccer", and "game" etc..

Here we use a machine learning model to learn the unit(including the query name)'s relevance score to the text content. The machine learned model incorporates both content and some page structural features:

1. Structural characteristics of unit u in a web page d, including the frequency of u in d's title, meta key words, outgoing another text, headings, table headers, table body, lists, etc;

2. Visual characteristics of u in d, including the frequency of u in d's visual title, relative font size, and when u is in bolding, italics, a changed color with respect to the context/background, etc;

3. General characteristics of d, including d's category, domains, document structures, language, length, url depth, etc;

4. Non-structural characteristics, including TFIDF score, offset, etc;

5. Corpus level characteristics, including document frequency, dominant category, etc.

- 1: Initialize $h_0(x) = \frac{\sum_{i=1}^N g_i}{N}$
- 2: for k = 1 to M (number of trees in gradient boosting) do
- 3: **for** i = 1 to N **do**
- 4: compute the negative gradient $r_{ik} = g_i h_{k-1}(x_i)$
- 5: end for
- 6: Fit a regression tree to $\{r_{ik}\}, i = 1, \dots, N$
- 7: giving terminal regions R_{jk} , $j = 1, ..., J_k$, J_k is the number of regions
- 8: for j = 1 to J_k do
- 9: compute

$$\tau_{jk} = \sum \frac{x_i \in R_{jk}(g_i = h_{k-1}(x_i))}{|i : x_i \in R_{jk}|},\tag{7}$$

- 10: average the residual in each terminal region.
- 11: end for
- 12: Update

$$h_k(x) = h_{k-1}(x) + \eta(\sum_{j=1}^{J_k} \tau_{jk} I(x \in R_{jk})),$$
(8)

where η is the shrinkage factor and I(.) in the indicator function.

13: end for

The Gradient Boosted Decision Tree (GBDT) [8,7] is used for content weighting with the above features. To train the machine learning relevance model, 1.3 million popular web pages are collected as the content weighting training data to learn the features and compute the frequencies. In addition, 400,000 query-url pairs are collected for manual annotation. We can get a web page from each url, and a query can be viewed as one of its features (or units). Annotators judge the relevance of a query to a web page on a 5-point scale (Perfect, Excellent, Good, Fair, Bad). Then the 5-point scales are scored as 1.0(Perfect) to 0(Bad), respectively.

Each element in the training data is written as $(x_i, g_i)_{i=1}^N$ (N is the size of training data) and we need to fit a function such that $g_i \approx h(x_i), i = 1, ..., N$. The lost function between g and h is

$$\sum_{i=1}^{N} |g_i - h(x_i)|^2 \tag{6}$$

We apply the gradient descent in functional space to minimize the discrepancy [13]. The algorithm of GBDT regression is as Algorithm 1:

There are two parameters, M and η . They are estimated with cross validation on the content weighting training set.

Here we have the relevance score $0 \le r(u) \le 1$ of a unit u (including the query name q). The higher value r(u) has, the more relevant the unit u is to the text content.

Since r(u) is the relevance of u to the text content, we can use it to estimate $v_r(u)$, which is the relevance of the unit u to the query name q.

$$v_r(u) = \begin{cases} 1 & \text{if } q = u \\ 0 & \text{if } r(q) < \theta \text{ and } r(u) < \theta \\ r(q)r(u) & \text{if } r(q) \ge \theta \text{ and } r(u) \ge \theta \end{cases}$$
(9)

where θ is a parameter and we get $\theta = 0.5$ from the training set.

Here we have two weight models $v_p(u)$ and $v_r(u)$, and they are both used to estimate the relevance of u to the query name q. We set the weights v(u) to be the maximum of $v_p(u)$ and $v_r(u)$.

$$v(u) = \max\{v_p(u), v_r(u)\}$$
(10)

For each unit u, we can get its relevance score v(u). All unit scores of a web page form a vector V. We can compute similarity between a pair of web pages through their vectors V.

3.3 Similarity Measures

Let V and V' be two the feature vectors of two web pages with the same entity name. Two types of similarity measures are used: (1) cosine similarity

$$Sim_{cos}(V,V') = \frac{V \bullet V'}{|V||V'|} \tag{11}$$

and (2) overlap similarity

$$Sim_{overlap}(V,V') = \frac{\sum_{u \in V, u \in V'} (v(u) + v'(u))}{\sum_{w \in V} v(w) + \sum_{w' \in V'} v'(w')},$$
(12)

where each u is one of the common units shared by V and V'; w and w' are all units in V and V', respectively. v(u), v'(u), v(w) and v'(w') are the scores of u. w and w' are computed on one of the two weighting models.

The performance of these two similarity measures is compared and presented in our experiment section. The results show that the disambiguation result based on overlap similarity is better than that of cosine similarity.

3.4 Clustering

and they satisfy

We employed the Hierarchical Agglomerative Clustering [9] algorithm to cluster documents with the same person name. Suppose C_i and C_j are two clusters. If there are two web pages V and V' in C_i and C_j , respectively,

$$V \in C_i, V' \in C_j,$$

 $Sim(V, V') > \gamma,$ (13)

 C_i and C_j will be merged into one cluster. Where Sim(V, V') can be computed with either Equation 11 or Equation 12. $\gamma = 0.25$ is tuned in the training sets.

Algorithm 2 pseudo-code of the clustering algorithm

1: $C = \{\{1\}, \{2\}, ..., \{n\}\}$ (*n* is the number of web pages) 2: $m \leftarrow n$ (*m* is the number of clusters) 3: while m > 1 do 4: $(C_i, C_j) \leftarrow \arg \max_{C_i, C_i \in C} Sim(C_i, C_j),$ where $Sim(C_i, C_j) = \max_{x \in C_i, y \in C_j} Sim(V_x, V_y)$ if $Sim(C_i, C_i) \le \gamma$ then go o 10 5: 6: $C_i \leftarrow C_i \cup C_j$ 7: $C \leftarrow C \setminus C_i$ 8: $m \leftarrow m - 1$ 9: end while 10: Output the clustering results

4 Experiments

In this section, we will evaluate our approach on the WePS datasets. The WePS-1 and WePS-2 datasets are used as the training and the test data for evaluation first. And our system's performance on the WePS-3 campaign is also presented.

4.1 Datasets

There are totally 76 query names in WePS-1. They are randomly selected from US Census, ambiguous person names in the English Wikipedia and program committee listing of a Computer Science conference [1]. For each query name, at most top 100 web pages returned by Yahoo! search engine are collected for disambiguation, so there are 6445 pages in total. In WePS-2, 30 query names are selected. Each of the query name has at most 150 pages from top search results, and there are 3444 web pages in total. In WePS-3 there are 300 query names, with top 200 web pages returned by Yahoo! for each query name, yielding 57355 evaluation pages in total. The WePS program committee asked annotators to manually label document clustering for each query name. The system's performance is measured by comparing the clustering generated from the algorithm with human labeled gold-standard test data. Two evaluation metrics are used in WePS: the Purity F-score and the B-Cubed F-score [2].

The WePS-1 datasets are used as the training data in our experiments to learn the similarity metrics and tune the parameters. The WePS-2 datasets are used as the test data. We submit our system's outputs on the WePS-3 datasets for this campaign, without any modification on the algorithm.

4.2 Experimental Results

In this sub-section, we will evaluate our approach through three aspects. First of all, the experimental results of the system under different feature sets and similarity measures are provided. Then our proposed two relevance weighting models will be evaluated. Finally we will show our system's performance on the WePS-3 datasets.

Features and Similarity Measures We have introduced in Section 3.1 that there are three types of features (or units): bag-of-words, named entities and Wikipedia concepts. Two similarity measures are also described in Section 3.3. Evaluation results about these two aspects are first presented, with TFIDF as the weight on each feature. The results with different feature sets and similarity measure are shown in Table 2.

| Similarity Measure | Co | sine | Overlap | | |
|---------------------------|-------------|--------------|-------------|--------------|--|
| Metric | F_{Cubed} | F_{Purity} | F_{Cubed} | F_{Purity} | |
| Bag-of-words | 0.69 | 0.72 | 0.71 | 0.74 | |
| Bag-of-words&Named Entity | 0.71 | 0.77 | 0.72 | 0.78 | |
| Bag-of-words&Wikipedia | 0.72 | 0.77 | 0.74 | 0.81 | |
| All | 0.72 | 0.78 | 0.73 | 0.80 | |

Table 2. Comparison of different features and similarity measures

 F_{Cubed} and F_{Purity} refer to B-Cubed F-score and Purity F-score, respectively. "Cosine" and "Overlap" mean cosine similarity and overlap similarity. "Bag-of-words&Named Entity" means both bag-of-words and named entity features are used. "All" means all the three types of features are used. From this table we can get four observations:

1. If we only use bag-of-words features, the results are not satisfactory. While combined with named entities and Wikipedia concept features, we can get better results;

2. Using all three types of features does not yield much better results than two types because there are many overlappings between named entities and Wikipedia concepts (a Wikipedia concept can also be viewed as a named entity);

3. The system based on overlap similarity outperforms the one based on cosine similarity;

4. The system can get the best results with overlap similarity and under bag-ofwords and Wikipedia features ($F_{Cubed} = 0.74$ and $F_{Purity} = 0.81$). Therefore, they are used to do the experiments in the following sub-sections.

Evaluation of the Weighting Models We will evaluate our proposed two relevance weighting models in this sub-section.

First, we evaluate the query relevance weighting model. There are three weighting functions: f_1 , f_2 and f_3 (see Equation 2 to Equation 4). The experimental results using these three functions as well as without using weighting functions are shown on the left-hand-side of Figure 1. "1", "2" and "3" are query relevance weighting functions f_1 , f_2 and f_3 , respectively. "0" means no weighting functions (only TFIDF weights). We can see from this figure that the performance of the system has been substantially improved with query relevance weighting functions. The system performs almost equally well under three functions.

We compare the system's performance under different weighting models. On the right-hand-side of Figure 1, "No Models" means we only use the TFIDF weight. "Query"



Query relevance weighting functions

Relevance weighting models

Fig. 1. Comparisons of different functions and models

means to use query relevance weighting model. The weighting function f_2 is used. "Content" stands for using content relevance weighting model. "Both" means both two weighting models. The figure shows marked improvement of performance when both of the two weighting models are used.

Results on the WePS-3 datasets In WePS-3 campaign, we evaluated our system on the WePS-3 test datasets. Table 3 shows our results. "Best" and "median" are the best and the median F_{Cubed} scores among all submissions, respectively. We have submitted three groups of results named "YHBJ-1", "YHBJ-2" and "YHBJ-3". YHBJ-1 and YHBJ-2 are both based on the extended feature sets including bag-of-words and Wikipedia concepts, with query relevance and content relevance weighting models. YHBJ-1 sets $\gamma = 0.3$ as the clustering threshold and YHBJ-2 sets $\gamma = 0.25$ (see Equation 13). YHBJ-3 does not use content relevance weighting model so its performance is lower than the other two submissions. From the table we can see that YHBJ-2 is the best result among all WePS-3 submissions.

| Submission ID | F_{Cubed} |
|---------------|-------------|
| Best | 0.55 |
| Median | 0.40 |
| YHBJ-1 | 0.52 |
| YHBJ-2 | 0.55 |
| YHBJ-3 | 0.50 |

5 Conclusion and Future Work

Our approach to web person name disambiguation extends existing bag-of-words features with Wikipedia concepts. In order to measure feature weights for calculating document clustering similarity, we employ two weighting models that take into account feature relevance to the query name and text content. Experiment results on the WePS-3 task 1 confirms the effectiveness of our method which outperforms all other competing algorithms.

In the future, we can make the following further improvement on this method:

1. There are no more than 200 top ranked web pages for each query name of the WePS datasets, but the large number of the rest of the search results contain a great deal of information which can help to get better clusters. We plan to build up a model to use the information returned by search engines as much as possible;

2. Currently, our distance weighting functions are applied to entities and Wikipedia concepts. We in the future plan to leverage semantic information in our weighting function;

3. Machine learning models can be used to calculate similarity scores in order to get more accurate estimation.

References

- 1. Javier Artiles, Julio Gonzalo, and Satoshi Sekine. The semeval-2007 weps evaluation: Establishing a benchmark for the web people search task. In *the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. ACL, June 2007.
- 2. Javier Artiles, Julio Gonzalo, and Satoshi Sekine. Weps 2 evaluation campaign: Overview of the web people search clustering task. In *WWW*, April 2009.
- 3. Ron Bekkerman and Andrew McCallum. Disambiguating web appearances of people in a social network. In *WWW*, pages 463–470, May 2005.
- Razvan Bunescu and Marius Pasca. Using encyclopedic knowledge for named entity disambiguation. In EACL, pages 17–24, April 2006.
- Silviu Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In EMNLP, pages 708–716, June 2007.
- 6. Ergin Elmacioglu, Yee Fan Tan, Su Yan, Min-Yen Kan, and Dongwon Lee1. Psnus: Web people name disambiguation by simple clustering with rich features. In *the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. ACL, June 2007.
- 7. Jerome Friedman. Greedy Function Approximation.
- 8. Jerome Friedman. Stochastic Gradient Boosting. Stanford University, 1999.
- 9. Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd Edition).* Springer, New York, 2009.
- Gideon S. Mann and David Yarowsky. Unsupervised personal name disambiguation. In *HLT-NAACL*, pages 33–40, May 2003.
- 11. Cheng Niu, Wei Li, and Rohini K. Srihari. Weakly supervised learning for cross-document person name disambiguation supported by information extraction. In *ACL*, pages 598–605, July 2004.
- Deepa Paranjpe. Entity-based cross-document coreferencing using the vector space model. In COLING-ACL, pages 79–85, August 1998.
- 13. Deepa Paranjpe. Learning document aboutness from implicit user feedback and document structure. In *CIKM*, November 2009.

12

- 14. Amanda Spink, Bernard J. Jansen, and Jan Pedersen. Searching for people on web search engines. *Journal of Documentation*, (60):266–278, 2004.
- 15. Xiaojun Wan, Jianfeng Gao, Mu Li, and Binggong Ding. Person resolution in person search results: Webhawk. In *CIKM*, pages 163–170, October 2005.
- 16. Minoru Yoshida, Masaki Ikeda, Shingo Ono, Issei Sato, and Hiroshi Nakagawa. Person name disambiguation on the web by two-stage clustering. In *WWW*, April 2009.
- 17. Minoru Yoshida, Masaki Ikeda, Shingo Ono, Issei Sato, and Hiroshi Nakagawa. Person name disambiguation by boostrapping. In *SIGIR*, July 2010.