Harvesting Related Entities with a Search Engine*

Shuqi Sun[1], **Shiqi Zhao**[2,1], **Muyun Yang**[1], **Haifeng Wang**[2], **and Sheng Li**[1]
[1]Harbin Institute of Technology, Harbin, China
{sqsun,ymy}@mtlab.hit.edu.cn, lisheng@hit.edu.cn
[2]Baidu, Beijing, China
{zhaoshiqi,wanghaifeng}@baidu.com

Abstract

This paper addresses the problem of related entity extraction and focuses on extracting related persons as a case study. The proposed method builds on a search engine. Specifically, we mine candidate related persons for a query person q using q's search results and the query logs containing q. The acquired candidates are then automatically rated and ranked using a SVM regression model that investigates multiple features. Experimental results on a set of 200 randomly sampled query persons show that the precision of the extracted top-1, 5, and 10 related persons exceeds 91%, 90%, and 84%, respectively, which significantly outperforms a state-of-the-art baseline.

1 Introduction

Facilitating efficient navigation in the knowledge space is essential to satisfying the current Web search demands. Named entities are vital building blocks of such a space, and retrieving related entities provides an efficient way of navigation. Related entity extraction refers to mining from text resources named entities with certain relationships between them, e.g. *person-affiliation* and *organization-location*. To this end, great efforts have been made recently in both academic (Banko et al., 2007; Wu and Weld, 2010) and industry (Zhu et al., 2009; Shi et al., 2010) circles.

A wide range of NLP applications could benefit from the high-quality repository of related entities. For query suggestion in Web search and e-business, given a query concerning some entity e, one can suggest entities related to e, in which users may also have interest. This could be regarded as a remarkable complement to the current techniques suggesting queries that merely contain the entity e or are similar in wording with e (Boldi et al., 2009). In online encyclopedia (e.g. Wikipedia) construction, linking together related entities can facilitate the users for effective navigation. Additionally, related entity extraction also allows us to automatically construct social networks.

In this paper, we focus on related person extraction, though our proposed techniques can be extended to other types of entities. Here, we provide a comprehensive definition of related persons, which fall into the following four categories.

- **Persons with definite relationships.** The relationships in this category can be explicitly represented with definite concepts, e.g. *parent*, *friend*, *colleague*, etc. Most previous literature focuses on such definite relationships between persons (Brin,1998; Etzioni et al.,2005; Banko et al.,2007; Zhu et al.,2009).

- **Persons related in certain events.** In the second category, the related persons interact with the query person in certain events, such as *co-starring* in the same movie or *co-authoring* in the same scientific paper.

- **Persons with similar identities** may also be of interest. In the case of query recommendation, for instance, when users query some particular type of persons, such as *actors* or *singers*, it is highly informative to recommend other similar persons of the same type.

- **Persons having other relationships** with the query person that do not fall into the three main categories above. For instance, given a person q, characters played by q (an actor) in a movie or created by q (an author) in her fiction belong to this category.

*This work was done when the first author was visiting Baidu.

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 1019–1027,
Chiang Mai, Thailand, November 8 – 13, 2011. ©2011 AFNLP

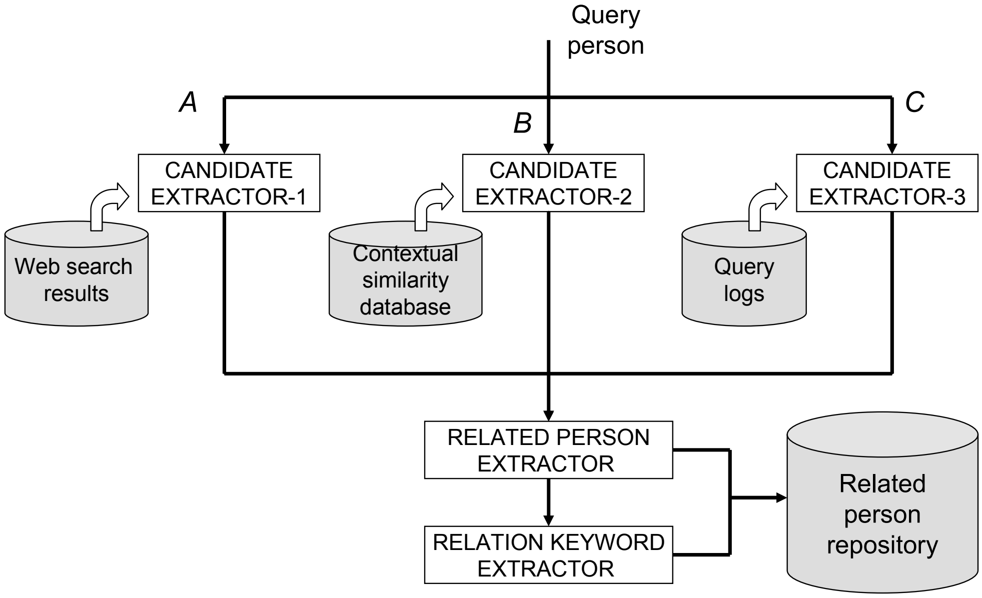

Figure 1: Overview of the proposed method.

Figure 1 illustrates the overview of our method. Our work is motivated by related entry suggestion in online encyclopedia construction and query suggestion in Web search. Thus, we are interested in finding related persons given a query person q. We employ a search engine SE to facilitate the extraction of related persons. In particular:

- We directly extract related persons from q's context in its search results returned by SE (flow A);
- Guided by distributional hypothesis (Harris, 1985), we mine q's related persons appearing in similar contexts (also collected from search results) with q (flow B);
- In addition, we exploit the query logs of SE, and extract q's related persons that co-occur with q in the same queries (flow C).

Extracting related persons aided by search engines has the following advantages. First, it can easily pinpoint web documents containing the query person, so that we can efficiently extract co-occurring persons and collect context information. Second, search results provided by search engines always contain the latest information, from which we can identify the newly emergent related persons. Third, the tremendous amount of search engines' indexed Web pages dramatically broadens the scope of the query person's context. Fourth, by exploiting search engine query logs, we can capture the related persons that the Web users are most interested in.

Using each of the three resources above, we aggregate up to 10 candidate related persons for each query person, and re-rank them with a Support Vector Machine (SVM) regression model that exploits multiple features, which finally outputs the top 10 related persons for each query person.

We evaluated our method with a set of 200 query persons randomly selected from Baidu[1] query logs, and compare it with Renlifang[2], a well known system developed for related person extraction. Experiment results show that our approach consistently outperforms Renlifang with a gap of 8%-13% in terms of averaged $Precision@K$. Specifically, 8.28% (0.915 vs. 0.845), 12.9% (0.9 vs. 0.797), and 9.73% (0.846 vs. 0.771) improvement has been achieved at rank 1, 5, and 10 respectively.

2 Related Work

Our work has its roots in both relation extraction and thesaurus construction. For relation extraction, the main approaches focus on binary relations between entities. These works use classifiers (Jiang and Zhai, 2007; Wang, 2008), extraction templates (Brin, 1998; Etzioni et al., 2005) or formulae (Agichtein and Gravano, 2000) to iden-

[1] http://www.baidu.com. Baidu is the largest commercial Chinese search engine in China.

[2] http://renlifang.msra.cn/

tify whether certain relations exist between pairs of entities. Supervised approaches view relation extraction as a classification problem, using either prior-designed feature sets (Jiang and Zhai, 2007) or kernel-based similarities (Wang, 2008) in classification. On the other hand, semi-supervised approaches avoid the heavy human labor in providing training examples by using bootstrapping techniques. For instance, DIPRE (Brin, 1998), Snowball (Agichtein and Gravano, 2000), and KNOWITALL (Etzioni et al., 2005) all adopt seed-pattern iterations to aggregate related entities.

Besides the works on traditional relation extraction, studies on open information extraction (Open IE) have emerged recently, which avoid pre-defining types of relations, and enjoy the capability of mining arbitrary types of semantic relations from both document collections (Shinyama and Sekine, 2006) and Web environment (Banko et al., 2007). Banko et al. (2007) build an Open IE system, TEXTRUNNER, that trains a Naïve Bayes classifier under the supervision of dependency rules. WOE (Wu and Weld, 2010) improves the precision and recall of TEXTRUNNER by mining clues from semi-structured texts in online encyclopedia and adopting different learning algorithms. Another descendent of TEXTRUNNER is the work of Mintz et al. (2009), which uses Freebase tuples as initial supervising information for training extractors. It is worth noting that building the learners is not mandatory. For example, Eichler et al. (2008) directly use syntactic patterns to perform Open IE.

There are also approaches that combine traditional relation extraction and Open IE together. Banko and Etzioni (2008) present H-CRF combining the two types of systems' output. StatSnowBall (Zhu et al., 2009) also performs both relation-specific extraction and Open IE. Like the technique proposed in (Banko and Etzioni, 2008), it formalizes the extraction problem as sequence labeling, but uses Markov Logic Networks (MLN) instead of Conditional Random Field (CRF).

In thesaurus construction, mainstream efforts related to our work consist of synonym / comparable entities clustering (Lin, 1998; Pantel, 2003; Wang and Cohen, 2007). Lin (1998) popularized the automatic clustering of similar words using distributional similarity. Pantel (2003) presents a more sophisticated clustering algorithm that first collects a small set of representative elements for each concept and then assigns words to their most-similar concept. Wang and Cohen (2007) alternatively investigate set expansion problem, that is, how to retrieve similar entities given a small number of seeds. With flexible matching patterns and random walk based ranking algorithm, their system outperforms Google Sets^{TM} in terms of Mean Average Precision (MAP). There are also researches forcusing on the relationship between verbs or adjectives, such as (Turney et al., 2003) and (Chklovski and Pantel, 2004).

In comparison to previous works on relation extraction, our work does not restrict itself to identifying pairs of entities whose relation is explicitly described by "infix"-like patterns, such as "Louis XVI *was born in* 1754". Alternatively, we mine related entities from context similarity and co-occurrence points of views. Moreover, through empirical studies, we find that query log is an effective data source in the extraction of related entities. On the other hand, different from synonym / comparable entities mining, our work retrieves a more extensive scope of results. In addition to entities similar or comparable to the query, we also extract those having more complicated relationships with the query, such as entities that interact with each other in certain events. Our work is also different from social network construction (Kautz et al., 1997), in that the evidences we use are also applicable to other types of named entities besides person.

3 Proposed Method

Our method for extracting related persons consists of two main stages. First, we generate candidate related persons in three fashions, based on context co-occurrence, contextual similarity, and query text co-occurrence, respectively. We collect up to 10 candidates from each source and combine them together. Second, we apply a SVM regression model to rate and rank the candidates and retain top 10 related persons for each query person. Five features are investigated, three of which are corresponding to the candidate extraction methods noted above, while the other two are based on joint-search of the query and each candidate.

3.1 Candidate Extraction

3.1.1 Context Co-occurrence

Co-occurrence is a traditional knowledge source for relation extraction (Church and Hanks, 1990).

In comparison to previous studies, we utilize a search engine to efficiently traverse the enormous Web corpus. In detail, we submit each query person q to a search engine and collect top 200 search results. We recognize co-occurred persons of q in the content of the search results within a window of limited length centered at each occurrence of q. The NER tool we used is based on a large NE table and a set of specific rules. In our experiments, the search engine we used is Baidu, and the length of the window is 5 words on both sides of the query. As a filtering step, we weed out persons that co-occur with the query for less than 3 times. Finally, we rank the co-occurred persons in descending order by their frequency, and keep up to top 10 as candidates.

3.1.2 Contextual Similarity

The distributional hypothesis presumes that words occurring in similar contexts tend to have similar meanings (Harris, 1985). In the scenario of related entity extraction, entities share similar contexts involve not only those with similar identities, e.g., two famous *pop stars*, but also those interacting with each other in the same event, e.g. two actors *co-starring* in the same movie. In this paper, we assemble in advance a large collection of persons, which contains approximately 160K person entities. For each person in the collection, we submit it to Baidu and extract its context words from its top 200 search results within the same text window (5 words on both sides) as above. The volume of the whole search result set is around 400GB In this manner, we generate a context word vector $v = (w_1, w_2, ..., w_K)$ for each person, in which w_i is a context word, and K is the total number of unique context words over the whole collection. The weight of w_i is calculated as:

$$W(w_i) = log(1 + tf_i) \cdot log(\frac{N}{qf_i}) \qquad (1)$$

where tf_i denotes the frequency of w_i in the context of the given person, qf_i denotes the number of persons in the collection whose context words contain w_i, and N denotes the total number of persons in the collection. We use logarithm on the frequency tf_i to reduce the influence of the words with extremely high frequency.

We extract candidate related persons for each query person q via selecting top 10 persons from the collection according to the contextual similarity with q. We compute the similarity between two context vectors v_i and v_j using Jensen-Shannon divergence (JSD), as it performs better than some other similarity computation methods, such as cosine similarity, in our experiments. We first normalize the input vectors by the sum of their components, and calculate the JSD as described in (Lee, 1999):

$$JSD(v_i, v_j) = \frac{1}{2}[KL(v_i \| v') + KL(v_j \| v')] \quad (2)$$

where KL denotes the Kullback-Leibler divergence between two vectors, and $v' = (v_i + v_j)/2$. Note that the larger the JSD is, the less similar two vectors are. Thus the similarity between vectors v_i and v_j is computed as $1 - JSD(v_i, v_j)$.

3.1.3 Query Text Co-occurrence

Web search queries represent the demand of information from the users. Queries are known to be noisy and of little syntactic structure. However, previous works have demonstrated that there is sufficient knowledge encoded in the query texts to perform information extraction tasks (Paca, 2007), and that extracting information within query logs can better represent the users' interests (Jain and Pennacchiotti, 2010).

We found from Baidu query logs that related persons are often searched together for users' curiosity about their relationships. Such pairs of persons consist of not only those with persistent relationships like *couples* and *friends*, but also those related in certain events, especially some hot news. In this spirit, we employ a Baidu query log containing approximately 9.08 billion raw queries to extract candidate related persons. For each query person q, we traverse the query log and extract persons that co-occur with q in the same queries. We filter out the persons that co-occur with q for less than N times (N is set 20 empirically in the experiments), and sort the left ones in descending order by the frequency of co-occurrence. Top 10 candidates are kept thereby.

3.2 Features for Regression

This paper recasts related person extraction as a regression problem. Given the candidates extracted as above, we train a regression model to score all the candidates and accordingly select the top-ranking ones as related persons. In this work, we investigate five features in the regression model.

Feature 1: Context Co-occurrence Feature (CCF). We design the first feature CCF to measure the co-occurring frequency of the query person q and a candidate related person c_j:

$$CCF(q, c_j) = \frac{Cooc(q, c_j)}{K} \tag{3}$$

where $Cooc(q, c_j)$ is the co-occurring frequency of q and c_j in the top K ($K = 200$) search results of q. Intuitively, the feature CCF is the average number of co-occurrences in each kept search result. We would like to stress that the feature computation is independent of candidate extraction, i.e., the feature CCF is available for all candidates extracted in the three manners above. This is also the case with the following features.

Feature 2: Contextual Similarity Feature (CSF). The contextual similarity described above is also taken as a feature. Given the context vectors v_q and v_j for the query person q and a candidate related person c_j, we devise the CSF feature as:

$$CSF(q, c_j) = 1 - JSD(v_q, v_j) \tag{4}$$

Feature 3: Query text Co-occurrence Feature (QCF). The QCF feature is designed to measure the frequency that the query person q and a candidate c_j are searched in the same queries. Here we define the QCF feature as the conditional probability of observing c_j in queries containing q:

$$QCF(q, c_j) = p(c_j|q) = n_{qj}/n_q \tag{5}$$

where n_q denotes the number of queries in the query log containing q, and n_{qj} denotes the number of queries containing both q and c_j.

In addition to the three features above, we also design two *joint-search* based features. Our motivation is that we can gather more clues about the relationship between two persons by searching them together and analyzing the search results. In practice, for each query person q and a candidate related person c_j, we form a joint-search query "q c_j" and submit it to Baidu. Roughly speaking, Baidu will first return results containing both q and c_j and then those containing either q or c_j. We keep top 200 search results and define the following two features:

Feature 4: Joint-search Co-occurrence Feature (JCF). A pair of related persons is supposed to co-occur in the joint-search results frequently. We thus use the JCF feature to measure the co-occurrence frequency of q and c_j in their joint-search results, which is defined as:

$$JCF(q, c_j) = \frac{2 \times s(q, c_j)}{s(q) + s(c_j)} \tag{6}$$

where $s(q)$ and $s(c_j)$ denote the numbers of sentences in the joint-search results that contain q and c_j respectively. $s(q, c_j)$ denotes the number of sentences containing both q and c_j.

Feature 5: Joint-search Distance Feature (JDF). The JDF feature takes the distance between q and c_j in the joint-search results into account. The underlying consideration is that related entities might appear closer to each other than irrelated ones. In practice, we only consider the cases in which q and c_j appear within the same sentences. The JDF feature is defined as:

$$JDF(q, c_j) = exp[-\frac{1}{S}\sum_{i=1}^{S} d_i(q, c_j)] \tag{7}$$

where S is the number of sentences in which q and c_j co-occur. $d_i(q, c_j)$ is their distance, i.e., the minimum number of words between them, in the i-th sentence they occur. We use the natural exponential function to restrict the range of the feature value.

3.3 SVM Regression Model

The judgment of the relatedness between persons is not binary. Closely related persons should receive more credit than loosely related ones. Thus we choose the regression scheme, which fits a continuous scoring function towards human annotated scores. We build SVM regression models using Gaussian kernel. The SVM toolkit we use in the experiments is SVM-Light v6.01[3], with its parameters at default setting. From the perspective of real application, for each new query person, we could generate its candidates as well as the features, and score them with the learned SVM regression model. However, in our experiments, we adopt 5-fold cross validation to validate the performance of the model. We will introduce the construction of the data sets in section 4.

[3] http://svmlight.joachims.org/

4 Experiments

4.1 Experimental Settings

4.1.1 Data Preparation

To construct the data set for model training and testing, we randomly sampled 200 Chinese query persons (i.e. queries that exclusively contain a single person entity) from the query logs of Baidu. For each query person, we extracted candidate related persons from the exploited resources as described in Section 3.1. In total, we acquired 4177 candidate related persons for the 200 sampled query persons, each of which has 20.9 candidates on average. We also obtained the top 10 related persons for the 200 queries produced by Renlifang for our comparison experiments. Renlifang is developed based on approximately 1 billion Web pages and object-level retrieval techniques (Nie et al., 2005; Nie et al., 2007; Nie et al., 2007), and is one of the most famous entity search engines in Chinese. Two native Chinese speakers were asked to rate all the candidates extracted by our approach as well as Renlifang results on a 4-point scale, i.e., 0,1,2,3. Specifically, a potential related person p of a query person q gets the rating 0 if p and q are not related. Ratings ranging from 1 to 3 correspond to relationships of different strengths, namely, *mild*, *moderate*, and *strong*. The raters were given instructions and examples that explained how to decide the relation strength between two persons. Two raters each labeled half of the data and checked the labeling results for each other. Those labeling results that had not reached an agreement would be discussed together, so as to generate a final rating.

4.1.2 Evaluation Metrics

To examine the performance of the SVM regression model, we randomly split the 200 query persons, along with their candidate related persons, in to 5 equal-size subsets, and performed 5-fold cross validation. In each run, four subsets are used for training and the other one is used for testing. The candidate related persons of each test person were automatically rated by the regression model, and top 10 of them were kept for evaluation. We adopt two metrics in the evaluation. In the first one, all the system outputs with a rating larger than 0 (i.e., 1,2,3) are counted as correct related persons of the test person q, without regard to the difference in relation strength. We calculate $Prec@K$ $(1 \leq K \leq 10)$ for the list of ranked related persons L_q of q and report the average $Prec@K$ on the whole data set:

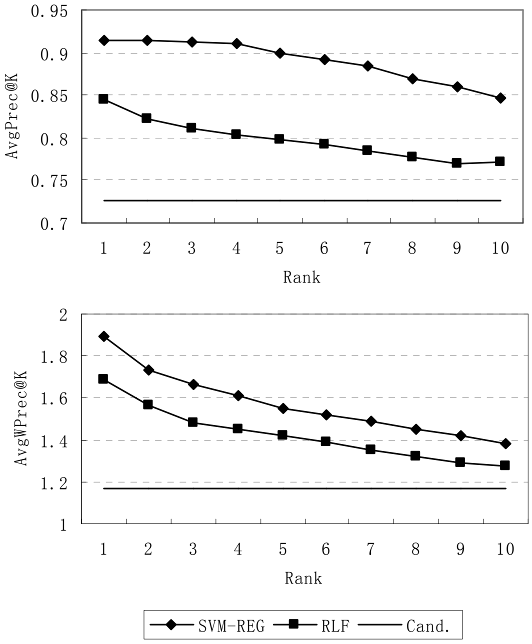

Figure 2: The comparison with Renlifang.

$$AvgPrec@K = \frac{1}{N} \sum_{N} \frac{\sum_{k=1}^{K} \mathbf{1}(R_{L_q}(k) > 0)}{K} \tag{8}$$

where $R_{L_q}(k)$ is the human-assigned rating of the k-th related person for q, $\mathbf{1}(\cdot)$ is the indicator function that yields 1 if $R_{L_q}(k) > 0$ and 0 otherwise, N is the total number of test persons that have at least one candidate related person extracted.

The second evaluation metric takes relation strength difference into consideration and assesses the system outputs based on an average weighted $Prec@K$, which is defined as:

$$AvgWPrec@K = \frac{1}{N} \sum_{N} \frac{\sum_{k=1}^{K} R_{L_q}(k)}{K} \tag{9}$$

4.2 Overall Comparison

In our experiments, we first compared the performances of our method and Renlifang according to $AvgPrec@K$ and $AvgWPrec@K$. The comparison results are depicted in Figure 2. In detail, the upper part of Figure 2 shows the performances of two systems in terms of $AvgPrec@K$. As can be seen, our method (SVM-REG) consistently outperforms Renlifang (RLF) by 8%-13%. Specifically, the performance gaps between these two

Rank	Related Entity (Translation)	Relationship
1	巩俐 (Gong Li)	Co-starring; Similar identity
2	姜文 (Jiang Wen)	Co-starring; Similar identity
3	葛优 (Ge You)	Co-starring; Similar identity
4	朱军 (Zhu Jun)	In certain event
5	吴宇森 (John Woo)	Actor-Director; In certain event
6	陈玉莲 (Idy Chan)	Love affair; In certain event
7	成龙 (Jackie Chan)	Comparative; Similar identity
8	钟楚红 (Cherie Chung)	Co-starring; Similar identity
9	刘德华 (Andy Lau)	Comparative; Similar identity
10	周星驰 (Stephen Chow)	Comparative; Similar identity

Table 1: Example of top-10 related persons for query person “周润发”.

methods are 8.28% (0.915 vs. 0.845), 12.9% (0.9 vs. 0.797), and 9.73% (0.846 vs. 0.771) at rank 1, 5, and 10. The comparison of $AvgWPrec@K$ (lower part of Figure 2) shows the same trend. At all ranks from 1 to 10, our approach significantly outperforms Renlifang by 9%-12%. Table 1 shows an example of the ranked results for query person “周润发” (Chow Yun-fat, Hong Kong actor).

To verify the effectiveness of the regression features, we carried out another series of experiments, eliminating one feature each time. The results are summarized in Figure 3. We can see that eliminating features CSF and JCF both result in a sharp decrease in the performance, which demonstrates the effectiveness of these two features. The performance also decreases when we ignore the QCF feature, but the drop is not evident. The other two features, namely CCF and JDF, seem useless in regression, since the performance is even slightly enhanced when they are eliminated.

Through observing the data, we find that the CCF feature seriously suffers from sparseness problem. In our experiments, only 2051 of the 4177 candidates have non-zero CCF value. Sparseness should be the main reason that invalidates the CCF feature. As to the JDF feature, throughout our analysis, there is no obvious correspondence between person relationship and their distance in the joint-search results, which means that the JDF feature is not discriminative.

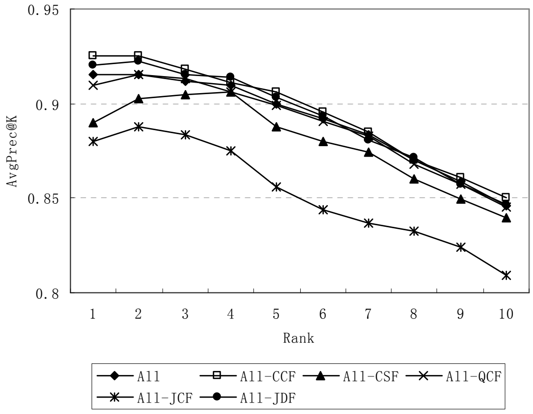

Figure 3: Evaluation of Feature Contribution.

	Con.	Sim.	Que.	All
# of test persons	196	200	178	200
# of candidates	1454	2000	1599	4177
# of cor. cands.	1229	1283	1359	3031

Table 2: Statistics of the candidate related persons extracted from three resources.

4.3 Contribution of Individual Resources

Recall that, in this work, the candidate related persons are acquired from three resources, based on contextual co-occurrence (Con.), contextual similarity (Sim.), and query text co-occurrence (Que.), respectively. It is therefore necessary to examine the contribution of each individual resource. Table 2 tabulates some statistics of candidate related persons extracted from three resources. Specifically, the first line of the table shows the number of test persons for which each resource can provide candidate related persons. The second line gives the total number of candidates yielded from each resource. The last line shows the number of correct candidates, namely, the candidates with ratings larger than 0.

We can find that each resource can provide a considerable number of candidate related persons. However, the qualities of the candidates differ a lot. In particular, the resource based on contextual similarity provides 10 candidate related persons for all the 200 test persons, but the precision is below 65%, which is the lowest. The other two

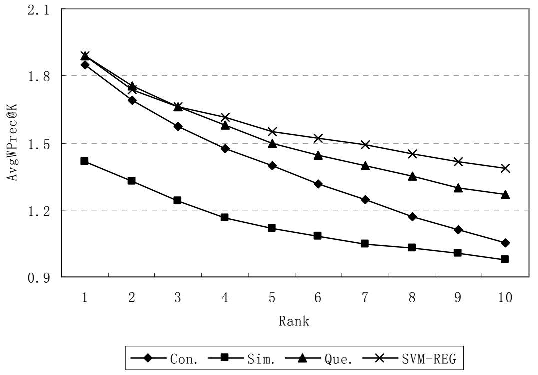

Figure 4: $AvgWPrec@K$ of ranked lists returned by individual resources and SVM regression model.

resources fail to provide candidates for some of the test persons, but their precisions are much higher, both of which exceed 84%. The last column of Table 2 shows the numbers of overall unique candidates and the correct ones. We can see that the overlap among three candidate sets is not large, indicating that all the three resources are contributing to the acquisition of candidate related persons.

To further investigate the quality, especially the relation strength, of the candidates acquired from each resource, we report $AvgWPrec@K$ of the raw ranked list of top 10 candidates from each resource in Figure 4. For the sake of comparison, we also include the $AvgWPrec@K$ results achieved by the SVM regression model using the whole feature set. The comparison results suggest that the $AvgWPrec@K$ scores of Que. and Con. come close to that of our SVM regression model when we only compare the top 3 or 4 candidates. However, the performance gap becomes larger as K grows. This is because the SVM regression model benefits from a much larger pool of candidates, from which it can select related persons of stronger relationships. We can also see that Sim. evidently underperforms Que. and Con., which is in accordance with the results reported in Table 2. In summary, the resource of Sim. assures the recall, while Que. and Con. provide relatively high-quality candidates, and the SVM regression model combines all evidences effectively.

5 Conclusions

In this paper, we make use of multiple resources provided by a search engine for acquiring related persons. The acquired candidates are rated and ranked with a SVM regression model that exploits various features. The following conclusions can be drawn from the experimental results:

First, the search engine facilitates the collection of needed resources in related entity extraction, with which we can easily collect ample web pages and query logs containing the queries of interest.

Second, the task of related entity extraction evidently benefits from the combination of multiple resources. We have observed significant improvement over the methods using each single resource.

Third, the SVM regression model is effective for rating and filtering the candidate related entities given discriminative features.

Our future work will be carried out along several directions. First of all, we will address the coreference resolution issue. The regression model will also be strengthened by employing more features. In addition, we will extend the method to other entity categories beyond person. We will also consider to extract cross-category related entities in the following work.

Acknowledgments

This work was supported by (1) China Postdoctoral Science Foundation (No.20100480100), (2) Beijing Postdoctoral Research Foundation, and (3) the Key Project of Natural Science Foundation of China (Grant No. 60736044). The authors are grateful for the anonymous reviewers for their valuable comments.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting Relations from Large Plain-text Collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, pages 85-94.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead and Oren Etzioni. 2007. Open Information Extraction from the Web. In *Proceedings of IJCAI*, pages 2670-2676.

Michele Banko and Oren Etzioni. 2008. The Tradeoffs Between Open and Traditional Relation Extraction. In *Proceedings of ACL-08:HLT*, pages 28-36.

Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, and Sebastiano Vigna. 2009. Query Suggestions Using Query-Flow Graphs. In *Proceedings of the 2009 Workshop on Web Search Click Data*, pages 56-63.

Sergey Brin. 1998. Extracting Patterns and Relations from the World Wide Web. In *WebDB Work-*

shop at 6th International Conference on Extending Database Technology, EDBT '98, pages 172-183.

Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. In *Proceedings of EMNLP*, pages 33-40.

Kenneth Ward Church and Patrick Hanks. 1990. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, 16(1):22-29.

Kathrin Eichler, Holmer Hemsen and Günter Neumann. 2008. Unsupervised Relation Extraction from Web Documents. In *Proceedings of LREC*, pages 1674-1679.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Dan-iel S. Weld, and Alexander Yates. 2005. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. *Artificial Intelligence*, 165(1): 91-134.

Zellig Harris. 1985. Distributional Structure. In Katz, J. J. (ed.), *The Philosophy of Linguistics*. New York: Oxford University Press. pages 26-47.

Alpa Jain and Marco Pennacchiotti. 2010. Open Entity Extraction from Web Search Query Logs. In *Proceedings of COLING*, pages 510-518.

Jing Jiang and Chengxiang Zhai. 2007. A Systematic Exploration of the Feature Space for Relation Extraction. In *Proceedings of HLT/NAACL*, pages 113-120.

Henry Kautz, Bart Selman, and Mehul Shah. 1997. Referral Web: Combining Social Networks and Collaborative Filtering. *Communications of the ACM*, 40(3): 63-65.

Lillian Lee. 1999. Measures of Distributional Similarity. In *Proceedings of ACL*, pages 25-32.

Dekang Lin. 1998. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of COLING-ACL*, pages 768-774.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant Supervision for Relation Extraction without Labeled Data. In *Proceedings of ACL-IJCNLP*, pages 1003-1011.

Zaiqing Nie, Yuanzhi Zhang, Ji-Rong Wen, and Wei-Ying Ma. 2005. Object-Level Ranking: Bringing Order to Web Objects. In *Proceedings of WWW*, pages 567-574.

Zaiqing Nie, Ji-Rong Wen, and Wei-Ying Ma. 2007a. Object-Level Vertical Search. In *Proceedings of the 3rd Biennial Conference on Innovative Data Systems Research*, pages 235-246.

Zaiqing Nie, Yunxiao Ma, Shuming Shi, Ji-Rong Wen, and Wei-Ying Ma. 2007b. Web Object Retrieval. In *Proceedings of WWW*, pages 81-90.

Patrick Pantel. 2003. Clustering by Committee. *Doctoral Dissertation, Department of Computing Science, University of Alberta.*

Marius Paşca. 2007. Organizing and Searching the World Wide Web of Facts - Step Two: Harnessing the Wisdom of the Crowds. In *Proceedings of WWW*, pages 101-110.

Shuming Shi, Huibin Zhang, Xiaojie Yuan, and Ji-Rong Wen. 2010. Corpus-based Semantic Class Mining: Distributional vs. Pattern-Based Approaches. In *Proceedings of COLING*, pages 993-1001.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive Information Extraction using Unrestricted Relation Discovery. In *Proceedings of HLT/NAACL*, pages 304-311.

Peter D. Turney, Michael L. Littman, Jeffrey Bigham, and Victor Shnayder. 2003. Combining Independent Modules to Solve Multiple-choice Synonym and Analogy Problems. In *Proceedings of RANLP*, pages 482-489.

Richard C. Wang and William W. Cohen. 2007. Language-Independent Set Expansion of Named Entities using the Web. In *Proceedings of ICDM*, pages 342-350.

Mengqiu Wang. 2008. A Re-examination of Dependency Path Kernels for Relation Extraction. In *Proceedings of IJCNLP*, pages 841-846.

Fei Wu and Daniel S. Weld. 2010. Open Information Extraction using Wikipedia. In *Proceedings of ACL*, pages 118-127.

Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. 2009. StatSnowball: a Statistical Approach to Extracting Entity Relationships. In *Proceedings of WWW*, pages 101-110.