LETTER

An Approach of Filtering Wrong-Type Entities for Entity Ranking

Junsan ZHANG[†a)], **Youli QU**[†], **Shu GONG**[†], **Shengfeng TIAN**[†], ***Nonmembers***, ***and*** **Haoliang SUN**[††], ***Student Member***

SUMMARY Entity is an important information carrier in Web pages. Users would like to directly get a list of relevant entities instead of a list of documents when they submit a query to the search engine. So the research of related entity finding (REF) is a meaningful work. In this paper we investigate the most important task of REF: Entity Ranking. The wrong-type entities which don't belong to the target-entity type will pollute the ranking result. We propose a novel method to filter wrong-type entities. We focus on the acquisition of seed entities and automatically extracting the common Wikipedia categories of target-entity type. Also we demonstrate how to filter wrong-type entities using the proposed model. The experimental results show our method can filter wrong-type entities effectively and improve the results of entity ranking.

key words: *related entity finding, entity ranking, type filtering, wikipedia*

1. Introduction

Traditional information retrieval system will return a list of documents when users submit an query. If users want to get answer entities, they need to manually find them from the retrieved documents. Related entity finding can automatically achieve the task of answer entities searching. According to TREC Entity track, the definition of related entity finding (REF) is: given a source entity, a relation and a target type, identify homepages of target entities that enjoy the specified relation with the source entity and that satisfy the target-type constraint [1]. REF provides a new way of information searching through entities. An example of a topic is given as follows:

< *num* > 17 < */num* >
< *entity_name* >The Food Network< */entity_name* >
< *entity_url* >clueweb09-en0006-55-17239
< */entity_url* >
< *target_entity* >Person< */target_entity* >
< *narrative* >Chefs with a show on the Food Network.
< */narrative* >

Entity ranking is an important issue of REF. How to filter wrong-type entities is a difficult task of entity ranking because the given target-entity types are too rough to exactly filter wrong entities. Some researchers try to refine the target-entity type making use of Wikipedia which provides plentiful category information of entities. M. Koolen et al. [2] proposed an approach which assigns a set of Wikipedia categories to the given target-entity types as new target categories. M. Bron at. [3] proposed an approach which manually assigned a number of initial Wikipedia categories for given target-entity type and searched continuously the sub-categories of initial categories as the target Wikipedia categories. M. Koolen [2]'s method and R. Kaptein [4]'s method both have some shortcomings: (1) it need manual participation and a set of topics have same target Wikipedia categories, (2) the filtering model only has two values. R. Kaptein et al. [4] proposed an approach using pseudo-relevance feedback of the top retrieved Wikipedia documents to automatically extract the categories of target type. Nevertheless, the number of target Wikipedia categories is insufficient in R. Kaptein [4]'s method which will lower the filtering results.

We propose an approach of filtering wrong-type entities: (1) it can automatically acquire the different target Wikipedia categories for each topic, (2) it gets more sufficient target Wikipedia categories, (3) it provides a filtering model which gives a cumulative score for each candidate entity. First, getting the target category name of each topic through parsing the narrative of topic. Second, we acquire seeds entities using query temples of hyponyms and extract their Wikipedia categories. Last, we automatically get the common categories of target-entity type and use type filtering model to filter wrong-type entities. The experimental result demonstrates that our approach can effectively filter wrong-type entities.

2. Methodology

2.1 Entity Ranking Model

According to the definition of REF, given a $Q(E^s, T, R)$, it returns a ranked list of relevant entities. In the paper, we use E^s to indicate the source entity, E^t indicate the target-entity, T indicate the target type, R indicate a relation between E^s and E^t. Using the conditional probability formula $P(E^t|Q)$ estimates REF task. Due to the $P(E^t|Q)$ is complex and difficult to estimating, we rewrite $P(E^t|Q)$ to:

$$P(E^t|Q) = \frac{P(E^t, Q)}{P(Q)} \tag{1}$$

Considering the denominator $P(Q)$ does not influence the

Manuscript received May 28, 2012.
Manuscript revised August 28, 2012.

[†]The author are with School of Computer and Information Technology, Beijing Jiaotong University, China.
[††]The author is with National Key Laboratory of Integrated Information System Technology, Institute of Software, Chinese Academy of Sciences, China.
a) E-mail: zhangjunsan@sina.com

DOI: 10.1587/transinf.E96.D.163

Copyright © 2013 The Institute of Electronics, Information and Communication Engineers

ranking of entities, we derive $P(E^t, Q)$ as follows:

$$\begin{aligned} P(E^t, Q) &= P(Q|E^t) \cdot P(E^t) && (2)\\ &= P(E^s, T, R|E^t) \cdot P(E^t) && (3)\\ &\propto P(E^s, R|E^t) \cdot P(T|E^t) \cdot P(E^t) && (4)\\ &= P(E^s, R, E^t) \cdot P(T|E^t) && (5)\\ &= P(R|E^s, E^t) \cdot P(E^s, E^t) \cdot P(T|E^t) && (6)\\ &= P(R|E^s, E^t) \cdot P(E^t|E^s) \cdot P(E^s) \cdot P(T|E^t) && (7)\\ &= P(R|E^s, E^t) \cdot P(E^t|E^s) \cdot P(T|E^t) && (8) \end{aligned}$$

We assume that type T is independent of source entity and relation R in (4). Assuming $P(E^s)$ is a uniform value in (7), we drop it. Now the ranking task is converted to three conditional probability question: $P(R|E^s, E^t)$, $P(E^t|E^s)$, $P(T|E^t)$. In this paper, our goal is to address the issue of wrong-type polluting entity ranking. So we only discuss $P(E^t|E^s)$and$P(T|E^t)$ in this paper.

2.2 Co-occurrence Model

We see $P(E^t|E^s)$ as a co-occurrence issue which expresses the association between source entity E^s and target-entity E^t. We use the formula [3] to estimate $P(E^t|E^s)$:

$$P(E^t|E^s) = \frac{Cooc(E^t, E^s)}{\sum_{E^{t'}} Cooc(E^{t'}, E^s)} \quad (9)$$

$E^{t'}$ indicates an entity co-occurrence with source entity E^s in documents. We use two approaches to estimate function $Cooc(E^t, E^s)$: (1) maximum likelihood estimate, (2) χ^2 hypothesis test [5].

Maximum likelihood estimate(MLE):

$$Cooc_{MLE}(E^t, E^s) = C(E^t, E^s)|C(E^s) \quad (10)$$

Where $C(E^t, E^s)$ indicates the number of documents in which E^t and E^s co-occur, $C(E^s)$ indicates the number of documents in which $C(E^s)$ occur.

χ^2 hypothesis test:

$$Cooc_{\chi^2}(E^t, E^s) = \frac{N \cdot (C(E^t,E^s) \cdot C(\overline{E^t},\overline{E^s}) - C(E^t,\overline{E^s}) \cdot C(\overline{E^t}, E^s))^2}{C(E^s) \cdot C(E^t) \cdot (N - C(E^t)) \cdot (N - C(E^s))} \quad (11)$$

Where $\overline{E^t}$, $\overline{E^s}$ indicate the E^t and E^s don't appears respectively, and N indicates the total number of documents. For example, $C(\overline{E^t}, \overline{E^s})$ expresses the number of documents in which both E^t and E^s don't appear.

2.3 Entity Filtering Model

The co-occurrence model preliminary ranks entities. But it can not resolve the problem of wrong-type entities polluting the ranking result. To address the issue of wrong-type entities will pollute the ranking result(i.e., $P(T|E^t)$. It represents how it is similar between target type and candidate entity type). We propose a type filtering model to deal with it as follows:

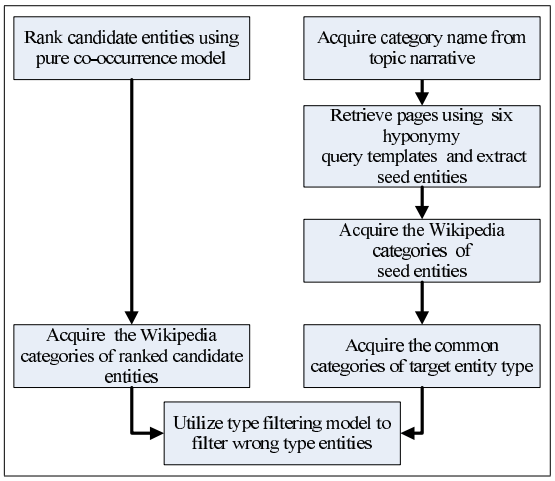

Fig. 1 The process of entity filtering.

$$P(T|E^t) = \frac{N(Cat(E^t) \cap Cat(T))}{N(Cat(T))} \quad (12)$$

Where $Cat(E^t)$ is the categories of a candidate entity E^t, and $Cat(T)$ is the categories of target type. $N(Cat(E^t) \cap Cat(T))$ indicates the number of categories which the candidate entity categories and the target categories co-own. $N(Cat(T))$ is the the number of categories which the target type owns. The process of entities filtering is shown in Fig. 1.

So it produces two issues:(1)how to get $Cat(E^t)$,(2)how to get $Cat(T)$. In this paper, we use Wikipedia pages as our sources. It is easy to acquire the categories $Cat(E^t)$ when we got a candidate entity E^t from Wikipedia pages. For example, we got a candidate entity "Bobby Flay" and we can acquire its categories Cat("*Bobby Flay*") through the url "http://en.wikipedia.org/wiki/Bobby Flay". However, it is difficult to get the accurate categories of target type $Cat(T)$ because the given target types are too rough. Such as, for entity track 2009, there are only three target types which are assigned: person, organization and product. Yet, we see the exact target type of each topic should be different through the topics' narratives. For example, there are two topics which have same target type (person). But they have completely different narratives: "Authors awarded an Anthony Award at Bouchercon in 2007", "Chefs with a show on the Food Network". From the narratives, the exact type which the former want is Authors but the latter want is Chefs. So if we can refine the target type according to the topic's narrative, it may filter wrong-type entity more accurately.

2.4 Acquiring the Common Categories of Target Type

First step, We extract the category name from topic's narrative with "Stanford Parser" [6]. For example, the narrative "Chefs with a show on the Food Network" is processed as follows:

(ROOT
(NP
(NP (NNS Chefs))
(PP (IN with)
(NP (DT a) (NN show)))
(PP (IN on)
(NP (DT the) (NNP Food) (NNP Network)))))
Commonly, the first noun phrase in the narrative is the category name which we want(For this example, it is "Chefs").

The next step, finding seed entities which are the hyponymy of the obtained category. We use hyponymy acquisition templates [7] to construct six query templates. For example "Chefs with a show on the Food Network", the query templates are:
(1)Chefs such as NP, (or/and) NP*
(2)such Chefs as NP, (or/and) NP*
(3)NP ,NP, or other Chefs*
(4)NP ,NP, or and other Chefs*
(5)Chefs , including NP, (or/and) NP*
(6)Chefs , especially NP, (or/and) NP*

Each query is submitted to a search engine(e.g. Google or Bing), retaining the top ten retrieved pages. We remove the html tags of the retrieved pages and acquire the sentences which contain the six hyponymy patterns. Using "Stanford Named Entity Recognizer" [6] processes the sentences and extract named entities. Now, we give an example of extracting seed entities using hyponymy query template *"Chefs such as NP,* (or/and) NP"*. The retrieved sentence is "chefs such as Wolfgang Puck and Sarah Molden". The seed entities "Wolfgang Puck" and "Sarah Molden" are extracted after the sentence is processed by NER.

Algorithm 1 Common categories acquisition.

```
define two array A,B and string variable C.
   A→deposit all seed entities categories.
   B→deposit the common categories.
   C→deposit a category of A .
put all seed entities categories into A.
for each C ∈ A do
   count the number of C appearing in A.
   if Count(C) > 1 then
      put C into B.
   end if
   take the first word W1 of C
   count the number of categories beginning with W1.
   if Count(categories beginning with W1) > 1 then
      put string structure beginning with "W1" into B.
   end if
   take the last word W2 of C
   count the number of categories ending with W2.
   if Count(categories ending with W2) > 1 then
      put string structure ending with "W2" into B.
   end if
end for
delete the repeated categories or category structures from B.
return B
```

The last step is to acquire the common categories of all hyponymy's categories. For a seed entity, if its Wikipedia page is not available or is a ambiguous Wikipedia page, drop it. We may acquire some seed entities and each entity has a number of categories. These categories can't be used directly. One aspect is because the seed entities' categories are incomplete, the other is the Wikipedia category structure is not a strict hierarchy and the category assignments are imperfect [8]. It is compulsory to seek out the common categories of the seed entities. We design Algorithm 1 to find the common categories of these seed entities.

For example, given three seed entities and their Wikipedia categories(each category is separated by semicolon):
(1)Alton Brown{1962 births;Food Network chefs;Living people;People from Marietta, Georgia}
(2)Wolfgang Puck{1949 births;American chefs;Living people;Austrian chefs}
(3)Giada De Laurentiis{1970 births;American television chefs;American television chefs;People of Campanian descent;Food Network chefs}

After the processing of common categories finding, the finally common categories $Cat(T)$ are:
(1)"Living people"
(2)"Food Network chefs"
(3)Ending with "births"
(4)Ending with "Chefs"
(5)Beginning with "People"
(6)Beginning with "American"

3. Experiment

3.1 Dataset and Evaluation Measures

In this paper, we utilize "The Lemur Project" which provides an online service of ClueWeb09 Category B as our corpus source and TREC2009 Entity Track's 15 topics as the test topics. For each topic, retrieving the top 1000 pages as the relevant documents. Since we restrict the scope of entities only in Wikipedia pages, only Wikipedia pages are reserved in the top 1000 retrieved pages. For the NER, we only consider anchor texts as entity occurrences. The precision and recall are our estimation measures. Using $P@10$ to express top 10 precision and $R@N$ express recall where N taken to be 100, 2000.

3.2 Experimental Result

We get the candidate entities from the retrieved wikipedia pages and the initial recall of relevant entities is shown in Fig. 2. The average entity recall of all topics is 90.78%. We utilize the pure co-occurrence model to rank candidate entities preliminarily. Then we use our proposed type filtering approach to filter wrong-type entities and rank candidate entities further. Figure 3 depicts the $P@10$ ranking result of which use pure co-occurrence model and add type filtering respectively. The line chart clearly demonstrates that wrong-type can be filtered and precision of top 10 effectively is im-

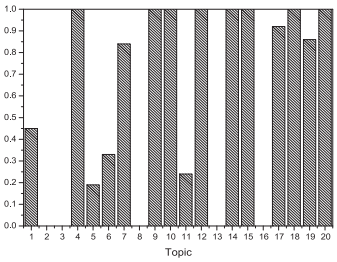

Fig. 2 The initial retrieval recall of relevant entities.

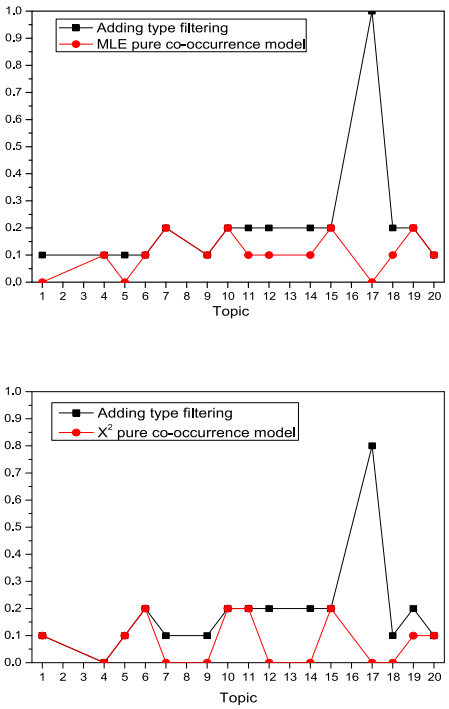

Fig. 3 Entity ranking result:P@10(using pure co-occurrence model and adding type filtering).

proved with our filtering method.

To show the effect of using type filtering model clearly, we take topic 17 as an example and list top 10 entities in Table 1. The relevant entities are indicated in bold. We see, using pure co-occurrence model (whatever it is MLE or χ^2) does't get relevant entities in the top 10. After adding type filtering, the wrong-type entities are effectively removed from the ranking list. The precision of top 10 achieves 100% when we use MLE in the co-occurrence model and 80% when we use χ^2 in the co-occurrence model.

Table 2 shows the average results of all topics using different measures. We see an increase in P@10, R@100 and R@2000 when using our proposed method filters wrong-type entities. It improved by 51.5% in P@10, 46.54% in R@100 when selecting MLE as the co-occurrence model and improved by 57.15% in P@10, 68.53% in R@100 when selecting χ^2 as the co-occurrence model. There are significant improvements both in P@10 and R@100. There is a little descent in R@2000 because some right entities may be

Table 1 Top 10 entities of topic 17. Relevant entities are indicated in bold.$Filter^{+MLE}$ indicates adding type filtering based on MLE model. $Filter^{+\chi^2}$ indicates adding type filtering based on χ^2 model.

Pure-MLE	$Filter^{+MLE}$	Pure-χ^2	$Filter^{+MLE}$
Food Network	**Bobby Flay**	The Food Network	James
Chef	**Mario Batali**	Intern	**Bobby Flay**
Main Page	**Rachael Ray**	Taste Page	Gordon Elliot
Port	**Alton Brown**	Chefs	**Mario Batali**
Television	**Michael Symon**	Food	**Rachael Ray**
Food	**Paula Deen**	Ingredient	**Alton Brown**
US	**Giada De Laurentiis**	Hour	**Michael Symon**
History	**Robert Irvine**	Eel	**Paula Deen**
Cooking	**Emeril Lagasse**	Gourmet	**Robert Irvine**
Iron Chef	**Cat Cora**	IGN	**Giada De Laurentiis**

Table 2 Results of all topics using different evaluation measures.

Method	P@10	R@100	R@2000
Pure-MLE	0.1067	0.3554	0.8943
$Filter^{+MLE}$	**0.22**	**0.6648**	**0.7893**
Pure-χ^2	0.08	0.1884	0.8588
$Filter^{+\chi^2}$	0.1867	0.5987	0.7356

Table 3 Comparison of our best results and other methods' results.

Method	P@10	R@100	R@2000
Our method	**0.22**	**0.6648**	**0.7893**
Bron [3]	–	0.5012	0.7881
Kaptein [4]	0.17	–	–
Koolen [2]	0.1550	0.4796	0.7081

filtered wrongly and it is unavoidable.

We also found that using proposed filtering method based on MLE co-occurrence model got a better result than based on χ^2 co-occurrence model. The improvement is significant in P@10(improved by 15.14%), R@100(improved by 9.94%) and R@2000 (improved by 6.80%). We make a comparison using our best results with other state-of-the-art methods' results and the results are shown in Table 3. We use author name to represent the proposed method and use "–" to indicate that this method doesn't employ this kind of evaluation metric. Table 3 shows our proposed method is the best at P@10,R@100 and R@2000.

The Bron's [3] approach also got remarkable results. It has two differences with our proposed approach: (1) the acquisition of its target Wikipedia categories is semi-automatic and a set of topics have same target Wikipedia categories. First, it must manually assign a set of Wikipedia categories to the given target types (Person, Product, organization). Then it continuously searches the sub-categories of the initial assigned Wikipedia categories (up to 10 levels). All categories(including the initial categories and the sub-categories) are regarded as the target Wikipedia categories. The procedure is shown in Fig. 4. The shortcomings of this way are: it is need a manual participation and a set of topics have same target Wikipedia-categories which can't reflect the diversity of each topic. By contrast, our proposed approach can automatically acquire the different target Wikipedia categories for each topic. (2) its filtering model only has two values (1 or 0). For example(assuming there are three candidate entities: E1, E2 and E3):

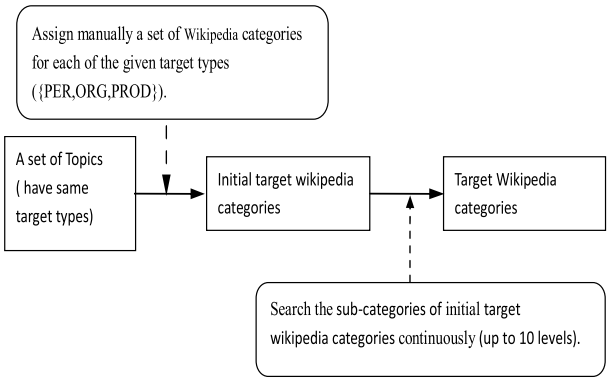

Fig. 4 The procedure of acquiring target Wikipedia categories in Bron's method.

The target Wikipedia categories are: A, B, C, D
The Wikipedia categories of E1 are: A, B, E
The Wikipedia categories of E2 are: G, H, M, N
The Wikipedia categories of E3 are: B, H, K

Although E1 has two same categories with the target categories and E3 has one. They have the same score (e.g. 1). E2's score is 0. Our filtering model (in Sect. 2.3) takes a different way of counting cumulative score. The score of entity E1 is 2/4 (because it has two same categories with the target categories), E3's score is 1/4 and E2's score is 0. We think the cumulative score can more effectively reflect how it is similar between the candidate-entity's categories and target Wikipedia categories.

The Koolen's [2] approach also need manually assign a set of Wikipedia categories to the given target types. But it doesn't continuously search the sub-categories. So the results are lower than Bron's. The Kaptein's [4] approach uses pseudo-relevance feedback of the top retrieved Wikipedia documents to automatically extract the categories of target type. It takes the top 10 feedback results and look at the 2 most frequently occurring categories. Nevertheless, its target categories are insufficient which lowers the filtering results and its results are lower than our resluts.

4. Conclusions

This paper investigates the problem of filtering wrong-type entities for entity ranking. The given target-entity type of topic is too rough to accurately filter wrong-type entities. We proposed an approach to automatically acquire the categories of target-entity types for each topic. Then we use proposed filtering model to remove wrong-type entities. The experiments show our approach can filter wrong entities effectively and greatly improve the entity ranking results.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities of China (Program No.2011JBM231)

References

[1] K. Balog, A.P. de Vries, P. Serdyukov, P. Thomas, and T. Westerveld, "Overview of the TREC 2009 entity track," Proc. TREC2009.
[2] R. Kaptein, M. Koolen, and J. Kamps, "Result diversity and entity ranking experiments," Proc. TREC2009.
[3] M. Bron and K. Balog, and M. de Rijke, "Ranking related entities: components and analyses," CIKM '10 Proc. 19th ACM International Conference on Information and Knowledge Management, pp.1079–1088, 2010.
[4] R. Kaptein, P. Serdyukov, A. De. Vries and J. Kamps, "Entity ranking using Wikipedia as a pivot," CIKM '10 Proc. 19th ACM International Conference on Information and Knowledge Management, pp.69–78, 2010.
[5] C.D. Manning and H. Schuetze, Foundations of Statistical Natural Language Processing, MIT Press, 1999.
[6] http://nlp.stanford.edu/software/index.shtml.
[7] M.A. Hearst, "Automatic acquisition of hyponyms from large text corpora," Proc. 14th Conference on Computational Linguistics, pp.539–545, 1992.
[8] J. Pehcevski, J.A. Thom, A.-M. Vercoustre, and V. Naumovski, "Entity ranking in Wikipedia: utilising categories, links and topic difficulty prediction," Information Retrieval, vol.13, no.5, pp.568–600, Jan. 2010.