An Adaptive Window-Size Approach for Expert-Finding

Fawaz Alarfaj, Udo Kruschwitz, and Chris Fox
School of Computer Science and Electronic Engineering
University of Essex
Colchester, CO4 3SQ, UK
{falarf, udo, foxcj }@essex.ac.uk

ABSTRACT

The goal of expert-finding is to retrieve a ranked list of people as a response to a user query. Some models that proved to be very successful used the idea of association discovery in a window of text rather than the whole document. So far, all these studies only considered fixed window sizes. We propose an adaptive window-size approach for expert-finding.

For this work we use some of the document attributes, such as document length, average sentence length, and number of candidates, to adjust the window size for the document. The experimental results indicate that taking document features into consideration when determining the window size, does have an effect on the retrieval outcome. The results shows an improvement over a range of baseline approaches.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Measurement, Performance, Experimentation.

Keywords

Expert-Finding, Entity Search, Adaptive Window, Proximity Search

1. INTRODUCTION

With the massive and ever-growing amount of electronic data, search engines have become crucial for any organisation that wants to help its employees with their day-to-day information needs. Traditionally, search engines, or information retrieval systems in general, function by returning a list of documents for the user's query, although the user's information need may not necessarily be in the form of documents. In fact, users more often search for specific things (people, organisations, or products) [9]. Many user information needs therefore, would be better answered by specific entities. Studies on user search behaviour show that entity search is the most prominent type of search on the web [5]. This led to the introduction of some entity search engines, such as product search (Google Product Search and Yahoo Shopping).

One special type of entity search is expert-finding. In expert-finding we are concerned with identifying experts who possess the relevant skills and knowledge on a given topic [1]. Today, expert-finding is considered an important task in the area of information retrieval, and it has attracted a great deal of attention and interest within the information retrieval community over the past few years [3]. People have different motives for seeking experts. Yimam-Seid and Kobsa [12] categorise these motives into two main groups, (i.e. expert finding and expert profiling). Firstly, in expert finding, users seeks expert as a source of information, where users are mostly interested in the question, 'Who knows about topic X?'. Secondly, in expert profiling, the motive is to find someone who can perform a given organisational or social function, where in this case users are equally interested in other questions; for example, 'How much does Y know about topic X?', 'What else does Y know?' or 'How is Y compared with others in his/her knowledge of X?'

Given a search topic, state-of-the-art expert-finding systems typically measure the knowledge of candidates from the textual content of top ranking documents, which are used to derive associations between candidates and search topics based on co-occurrences [7, 3]. The co-occurrence of candidate identifiers with query terms is considered to provide evidence of expertise. In addition, the nature and frequency of co-occurrences is used in estimating the probability of a person being an expert. The general assumption is that the more often a candidate is found in a document containing many terms describing the topic, the more likely he or she will be an expert on this topic. The second assumption is that the closer the candidate identifiers are to the query terms, the stronger the association between them. Using these assumptions, some studies consider the proximity of query terms and candidate identifiers using fixed-size windows. Zhu *et al.* tested 31 window sizes on the W3C collection[1] ranging from 5 to 1100. They found the best window size to be around 200 words. According to Zhu *et al.*, small window sizes could lead to high precision, but low recall. On the other hand, large window sizes lead to high recall, but low precision [14]. Some studies therefore, consider multiple

[1]The same collection is used in this paper.

DIR 2013, April 26, 2013, Delft, The Netherlands.

levels of associations in documents by combining multiple fixed window sizes [14, 2].

In this paper, we consider the idea of an *adaptive* window size, where the size of the window is a function of various document features. We argue that each document has distinct features that differ from other documents in the collection. Using these features to set the window size could improve the overall ranking function. There are many document features that could be examined. We focus on three of them: document length, average sentence length, and candidate frequency (i.e. the number of candidates that appear in a document). To the best of our knowledge, no existing work has dealt with using the document features to determine the optimal window size for the proximity function.

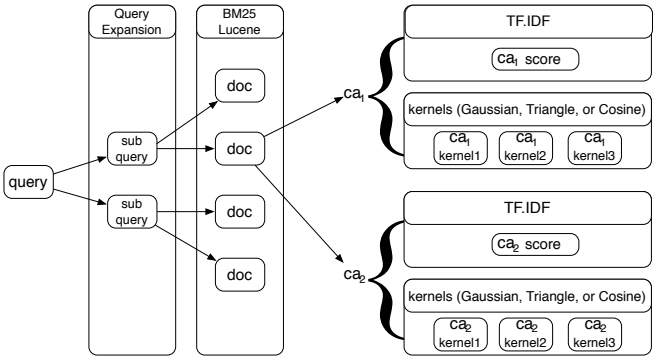
It is important to note that the adaptive window size approach could be applied to any proximity search, in particular for an entity-oriented search, a generalisation of expert search. We carried out the study in the expert search domain due to the availability of an expert-search benchmark.

The main research question considered here is whether an adaptive window size leads to improvements over fixed window size methods.

2. EXPERT-FINDING FRAMEWORK

As described above, the input for any expert-finding system is the user's submitted query. This query then could be normalised and different query expansion techniques could be applied to it. Next, the query is passed to an underlying search engine; in this work, we used Lucene[2] as our search engine, with a BM25 ranking function. For each query, only the top 100 documents returned by the search engine were considered. We used these documents to rank the candidates based on two measures. First, based on their frequency occurrence, and second, based on the proximity between the candidate's evidence and the query occurrences in the document (Figure 1).

Figure 1: Expert-Finding Framework



3. ADAPTIVE WINDOW SIZE FOR PROXIMITY RANKING

Proximity approaches have been successfully used in different applications, which enhance the quality of the retrieval systems. In particular, the work of Petkova & Croft, [11] directly addresses the use of a kernel for proximity of name / query term occurrences in expert search. The other works, among others, which examine proximity in expert search include Macdonald *et al.* [6] and Petkova & Croft [10].

In this work, the window size for the proximity function will be determined for each document based on the following features.

(1) Document Length: According to Miao *et. al.* [8], in large documents, it is more likely to find more occurrences of a query topic. It is also more likely to have irrelevant words (noise) in such documents. Thus, in order to minimise the negative influence of noise, the window size should be relatively smaller as the document gets bigger. **(2) Candidate Frequency**: This term is used to refer to the number of candidates found in a document. When a document has more occurrences of candidates' evidence, the window size should be relatively larger to accommodate more occurrences. **(3) Average Sentence Length**: The window size is adjusted in proportion to the average sentence length (in tokens) in the document. We combine these features in the following equation:

$$\begin{aligned} &Window\ Size = \\ &\frac{\sigma}{3} * (\log(\frac{1}{DocLength}) * \beta_1 \\ &\quad + CanFreq * \beta_2 \\ &\quad + AvgSentSize * \beta_3) \end{aligned} \tag{1}$$

σ is a variable that allows to scale the window size. We explore a wide range of values for σ, (see below). The β weighting factors, which determine each feature's contribution in the equation, have been set empirically, where $\sum_{i=1} \beta_i = 1$. The TREC2005 data includes ten training topics[3]. We used these topics to train our β variables, thus having a clear distinction between test and training data.

Although the proposed model used the three features, we will also report experiments for each feature individually.

After establishing the size of the window, it is applied to every full match for the query found in the document. Then, the candidate evidence neighbouring this term is extracted; each one within the window will be given a weight depending on its distance from the query.

The advantage of this window is that it provides a graded proximity boost. Candidates with an index close to the query terms will receive the highest boost. As the candidate indexes drift further and further away, the boost will gradually decrease until it reaches the end of the window. A document can contain multiple query terms. In this case, we place a window at each occurrence. If, for example, a document has two query terms, two windows are placed, but centred at different locations. If the two windows are close to each other, both windows could boost candidates that appear between them.

Three different kernel functions were used to calculate the weight: Gaussian, Triangle, and Cosine [13].

4. EXPERIMENTS

To evaluate our approach, we used the document collection of the W3C corpus and the test sets of the 2005 TREC Enterprise track. The W3C corpus includes a predefined

[2] `http://lucene.apache.org/core/`

[3] `http://trec.nist.gov/data/enterprise/05/ent05.expert.trainingtopics`

Run		σ	MAP	r-prec	bpref	P@5	P@10	P@20
Baseline		N/A	0.1532	0.2531	0.2749	0.3210	0.2519	0.1908
Gaussian	baseline	N/A	0.3001	0.3554	0.4297	0.5092	0.3595	0.3089
		350	0.3363	0.3808	0.4787	0.5200	0.3900	0.3350
		400	0.3342	**0.3975**	0.4737	0.5200	0.4000	0.3300
		450	0.3454	0.3955	**0.4954**	0.5200	0.4099	**0.3450**
		500	**0.3454**	0.3905	0.4861	0.5200	0.4199	0.3350
		550	0.3443	0.3905	0.4890	0.5200	**0.4299**	0.3400
		600	0.3402	0.3905	0.4851	0.5200	0.4199	0.3350
		650	0.3357	0.3821	0.4792	0.5200	0.4099	0.3350
Triangle	baseline	N/A	0.2358	0.3331	0.3602	0.4023	0.3329	0.2750
		350	0.3126	0.3642	0.4494	0.4800	0.4099	0.3199
		400	0.2974	0.3509	0.4427	0.4800	0.4099	0.3199
		450	**0.3261**	0.3793	**0.4623**	0.5199	**0.4299**	**0.3300**
		500	0.3169	**0.3804**	0.4330	0.5600	0.4200	0.3050
		550	0.3144	0.3776	0.4209	0.5600	0.4099	0.3050
		600	0.3036	0.3767	0.4093	**0.5800**	0.3800	0.2950
		650	0.2836	0.3490	0.3869	0.5400	0.3900	0.2800
Cosine	baseline	N/A	0.2700	0.3605	0.4078	0.4102	0.3495	0.3095
		350	0.2735	0.3557	**0.4494**	0.4219	0.3999	0.3499
		400	0.2757	0.3414	0.3149	0.4191	0.4199	0.3599
		450	0.2761	0.3498	0.3149	0.4191	0.4199	0.3599
		500	**0.2811**	0.3639	0.3199	**0.4241**	**0.4399**	**0.3599**
		550	0.2800	0.3639	0.3199	0.4232	0.4399	0.3599
		600	0.2756	0.3639	0.3149	0.4155	0.4399	0.3599
		650	0.2744	0.3639	0.3149	0.4155	0.4199	0.3599

Table 1: The performance of the Adaptive Window-size Approach for different proximity functions. Highest scores for each category are typeset in boldface. The best run overall are typeset in boldface and underlined.

list of 1092 candidates, 331,037 documents, and 50 topics, each of which is provided with a relevance judgement. We selected this collection in order to test our method on a simple and most basic form of expert-finding[4].

We removed stopwords and HTML markup, and treated all documents as plain text. For evaluation, we applied a range of standard IR measures, but in our discussion we focus on Mean Average Precision (MAP).

In this work, we use the two-stage model for the initial candidate ranking by calculating the probability of the candidate given the query, $P(ca|q)$, as follows:

$$P(ca|q) = \sum_d P(d|q) \cdot P(ca|d) \tag{2}$$

where $P(d|q)$ is the document relevance to the query, which is calculated by the underlying search engine, and $P(ca|d)$ is the candidate's probability given the document. In our baseline, $P(ca|d)$ is calculated using the full document without a proximity function. Whereas in all other experiments, we apply Equation 1 to find the optimal window size for the current document. The proximity functions will only consider the occurrences within this window of text.

Our first baseline is a frequency-based approach. In this baseline, a $TF - IDF$ weighting scheme is used in order to obtain the candidate's importance in a particular document,

[4]Other forms of expert finding include finding similar experts and finding all expertise for a given candidate.

Feature	CanFreq	AvgSentSize	DocLength
Best σ value	250	600	450
MAP	0.2806	0.2798	0.2777
bpref	0.3452	0.3269	0.3452
r-prec	0.4147	0.4199	0.4112
P@5	0.4199	0.4189	0.4199
P@10	0.3599	0.3499	0.3499
P@20	0.3100	0.3100	0.3050

Table 2: The performance of the Adaptive Window-Size Approach using a single feature. Only the best result for each feature is reported.

while at the same time integrating it with the candidate's general importance [2]:

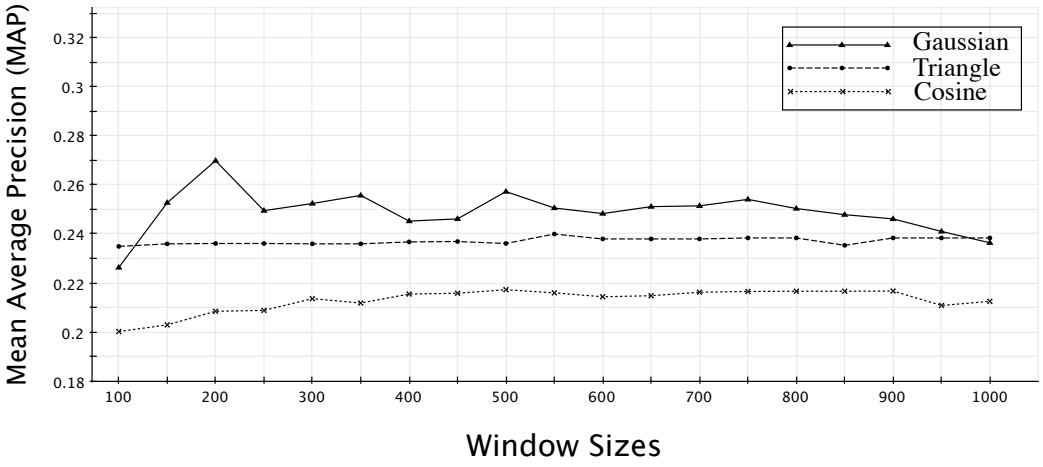
$$P(ca|d) = \frac{n(ca,d)}{\sum_{ca'} n(ca'd)} \cdot \log \frac{|D|}{|\{d' : n(ca,d') > 0\}|} \tag{3}$$

where $n(ca, d)$ is the number of times the candidate ca appears in the document d and $|D|$ is the total number of documents in the collection.

Starting from the baseline, we used the proximity functions with adaptive window size to boost the relevance score.

To test the effect of each document feature separately, we first generated an adaptive window size with only one feature and used it with a Gaussian proximity function. In Table 2, we report the best runs for each feature separately (i.e. CanFreq with $\sigma = 250$, AvgSentSize with $\sigma = 600$, and DocLength with $\sigma = 450$).

Figure 2: MAP for fixed window sizes



We used our adaptive window-size method (Equation 1), with the three proximity functions at different σ values ranging from 0 to 1000 with an increment of 50. We only report the results for σ values between 350 and 650. The results below 350 and above 650 drop gradually, so they were not reported. Furthermore, we calculate a baseline for each proximity function. In this baseline, we set the window size to be equal to the document length. Our results are summarised in Table 1.

The top MAP of 0.3454 is achieved using a Gaussian proximity function with an adaptive window size where $\sigma = 500$.[5] We found that the difference between our best run and the baseline is statistically significant (using paired t-tests on average precision values at $p < 0.05$). Moreover, we found that the differences between the best run for each proximity function and its baseline were also statistically significant.

For comparison, we used a range of fixed window sizes. We calculated MAP for fixed windows in a range from 100 to 1000 in increments of 50. We repeated the experiments using the three proximity functions (Gaussian is shown to be significantly better than the other two functions, with a top result of MAP=0.27 at a window size of 200); see Figure 2.

5. CONCLUSIONS

We introduced the idea of an adaptive window size for expert-finding. Thus, for the proximity function, the size of the window will be set based on current document features rather than a fixed window for all documents in the collection. Adopting this method results in significant improvements over standard metrics. This is true for all proximity functions used in this study (i.e. Gaussian, Triangle, and Cosine). We found that the best results were achieved using a Gaussian function. As for future work, we plan to investigate the effectiveness of using other document features such as the readability index for determining the optimal window size. We also plan to test the adaptive window size method on other expert-finding collections and also on other TREC benchmarks.

[5]For comparison, the best run at TREC 2005 reported a MAP value of 0.2749 [4], but do note that this was in 2005.

6. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval: the concepts and technology behind search.* Addison-Wesley, Pearson, 2ed edition, 2011.

[2] K. Balog, L. Azzopardi, and M. de Rijke. A language modeling framework for expert finding. *Inf. Process. Manage.*, 45(1):1–19, Jan. 2009.

[3] K. Balog, Y. Fang, M. de Rijke, P. Serdyukov, and L. Si. Expertise retrieval. *Foundations and Trends in Information Retrieval*, 6(2-3):127–256, 2012.

[4] N. Craswell, A. de Vries, and I. Soboroff. Overview of the TREC-2005 enterprise track. In *TREC 2005 Conference Notebook*, pages 199–205, 2005.

[5] B. Jansen and A. Spink. How are we searching the world wide web? a comparison of nine search engine transaction logs. *Information Processing & Management*, 42(1):248–263, 2006.

[6] C. Macdonald, D. Hannah, and I. Ounis. High quality expertise evidence for expert search. *Advances in Information Retrieval*, pages 283–295, 2008.

[7] C. Macdonald and I. Ounis. Searching for expertise: Experiments with the voting model. *The Computer Journal*, 52(7):729–748, 2009.

[8] J. Miao, J. X. Huang, and Z. Ye. Proximity-based rocchio's model for pseudo relevance. SIGIR '12, pages 535–544, Portland, Oregon, 2012.

[9] G. Mishne and M. de Rijke. A study of blog search. *Advances in information retrieval*, pages 289–301, 2006.

[10] D. Petkova and W. B. Croft. Hierarchical language models for expert finding in enterprise corpora. pages 599–608, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[11] D. Petkova and W. B. Croft. Proximity-based document representation for named entity retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 731–740, New York, NY, USA, 2007.

[12] D. Yimam-Seid and A. Kobsa. Expert-finding systems for organizations: Problem and domain analysis and the demoir approach. *Journal of Organizational Computing & Electronic Commerce*, 13(1), 2003.

[13] J. Zhao, J. X. Huang, and B. He. CRTER: using cross terms to enhance probabilistic information retrieval. SIGIR '11, pages 155–164, Beijing, China, 2011.

[14] J. Zhu, D. Song, and S. Rüger. Integrating multiple windows and document features for expert finding. *J. Am. Soc. Inf. Sci. Technol.*, 60(4):694–715, Apr. 2009.