Extracting Relevant Snippets from Web Documents through Language Model based Text Segmentation [*]

Qing Li
Comp. Sci. and Eng. Dept.
Arizona State University

K. Selçuk Candan
Comp. Sci. and Eng. Dept.
Arizona State University

Yan Qi
Comp. Sci. and Eng. Dept.
Arizona State University

Abstract

Extracting a query-oriented snippet (or passage) and highlighting the relevant information in long document can help reduce the result navigation cost of end users. While the traditional approach of highlighting matching keywords helps when the search is keyword oriented, finding appropriate snippets to represent matches to more complex queries requires novel techniques that can help characterize the relevance of various parts of a document to the given query, succinctly. In this paper, we present a language-model based method for accurately detecting the most relevant passages of a given document. Unlike previous works in passage retrieval which focus on searching relevance nodes for filtering of preoccupied passages, we focus on query-informed segmentation for snippet extraction. The algorithms presented in this paper are currently being deployed in OASIS, a system to help reduce the navigational load of blind users in accessing Web-based digital libraries.

1. Related Work

Many search engines treat documents atomically in indexing, search, and visualization. When searching the web for a specific piece of information, however, long documents returned as atomic answers may not be always an effective solution: first, a single document may contain multiple relevant parts; secondly, a large number of long documents returned as matches increase the overhead on the user while sifting through possible matches. While the traditional approach of highlighting matching keywords [2] helps when the search is keyword oriented, this approach become inapplicable when other query models (which may take into account users preferences, past access history, and context - such as ontologies) are leveraged to increase the search precision. We note that extracting a query-oriented snippet (or passage) and highlighting the relevant information in a given document can help reduce the result navigation cost of end users. This is especially true when the number of relevant documents is large; the documents involved are long and contain many instances of the query keywords, or when the users have difficulty accessing complete documents in search for relevant parts [1]. Thus, in many contexts, there is a strong need for finding appropriate snippets to represent matches to complex queries and this requires novel techniques that can help characterize the relevance of parts of a document to the given query succinctly.

The fundamental challenge in finding relevant snippets is to identify coherent, query-oriented components in the document. A simple and coarse way to identify snippets is to extract a query-oriented passage from well-delineated areas, where the keyword phrases is found in the document [1, 4], for example, rely on structural markups to identify passage boundaries. Google also extracts snippets from any one or combination of predefined areas within web pages [6]. While this approach may be applicable to the well delineated pages, it essentially side steps the major critical challenge that would need to be solved before developing a solution that can be applied to all types of web documents: “how to locate the boundaries that separate the relevant snippet from the rest of the document?” . Due to this variable length characteristic of query-oriented snippets, a window-based extraction [1] is not satisfactory, except maybe for very high-level quick-and-dirty filtering support. A more content-informed approach involves identifying coherent segment in a given document through pre-

[*] This work is supported by DES/RSA grant DES060664-1 “Ubiquitous Environment to Facilitate Access to Textbooks and Related Materials for Adults and School Age Children who are Blind or Visually Impaired” and NSF ITR Grant ITR-0326544 “iLearn: IT-enabled Ubiquitous Access to Educational Opportunities for Blind Individuals”.

[1] This is true when users are accessing the system through mobile devices. Another user population which suffers from long documents include blind and visually impaired users. When accessing digital documents, these users have to rely on screen reader software, such as JAWS, to read the document line by line, which is time consuming and makes search inconvenient. As part of our Organizing, Annotating, and Serving Information to individuals without Sight (OASIS) project [7], we are developing assistive technologies to help students, who are blind and visually impaired, access digital documents and Web pages.

processing [3, 8] and then filtering segments through post-processing. While this approach enables identification of coherent segments matching a given query, since the segments are identified through pre-processing, the boundaries are not informed of the query-context.

In this paper, we present a relevance language model based method for accurately detecting the most relevant passages of a given document.

2. RELEVANCE LANGUAGE MODELS

The language modeling framework for information retrieval was first introduced by Ponte and Croft [9] and marked a departure from traditional models of relevance. A language model is a probability distribution that captures the statistical regularities (e.g. word distribution) of natural language use [11].

A document d, also represented as a bag of words $(w_1, w_2...w_k)$, consists of two parts: a snippet and a non-snippet part. In particular, a snippet is a contiguous sequence of words in the document. In this work, we start with the assumption that there is an (unknown) relevance model, which captures the probability distribution of words for query-relevant snippets for a given query. That is, a query-oriented snippet of a given document can be generated with a hitherto unknown relevance model, M_r. Furthermore, the non-snippet part of the document can be generated by an irrelevance model, $M_{\bar{r}}$. M_r ($M_{\bar{r}}$) captures the probability,$P(w|M)$, of observing a word in a document that is relevant (irrelevant) to the query-oriented summary.

For the irrelevance model, estimating probability, $P(w|M_{\bar{r}})$, of observing a word, w, in a document that is irrelevant is relatively easy: for a typical query, almost every document in the collection is irrelevant. Therefore, irrelevance (or background) model can be estimated as follows:

$$P(w|M_{\bar{r}}) = \frac{cf_w}{coll.size} \tag{1}$$

where cf_w is the number of times occurs in the entire collection, and $coll.size$ is the total number of tokens in the collection.

Estimating the probability, $P(w|M_r)$, of observing a word, w, that is relevant to a query-oriented snippet is harder. Of course, if we were given all the relevant snippets from all the documents, the estimation of these probabilities would be straightforward. However, in a typical setting, we are not given any examples of the relevant snippets. Lavernko and Croft [5], propose to address this problem by making the assumption that the probability of a word, w, occurring in a document (in the database) generated by the relevance model should be similar to the probability of co-occurrence (in the database) of the word with the query keywords. More formally, given a query, $q = \{q_1q_2...q_k\}$, we can approximate $P(w|M_r)$ by $P(w|q_1q_2...q_k)$. Note that, in this approach, queries and relevant texts are assumed to be random samples from an underlying relevance model, M_r, even though in general the sampling process for queries and summaries could be different.

In our work, to construct the relevance model M_r, we build on [5] as follows:

- We use the query, q, to retrieve a set of highly ranked documents R_q. This step yields a set of documents that contain most or all of the query words. The documents are, in fact, weighted by the probability that they are relevant to the query.

- As with most language modeling approaches, including [9], we calculate the probability, $p(w|d)$, that the word, w, is in a given document, d, using a maximum likelihood estimate, smoothed with the background model:

$$\begin{aligned} P(w|d) &= \lambda P_{ml}(w|d) + (1-\lambda)P_{bg}(w) \\ &= \lambda \frac{tf(w,d)}{\sum_v tf(v,d)} + (1-\lambda)\frac{cf_w}{coll.size} \end{aligned} \tag{2}$$

  Here, $f(w, d)$ is the number of times the word, w, occurs in the document, d. The value of, λ, as well as the number of documents to include, R_q, are determined empirically.

- Finally, for each word in the retrieved set of documents, we calculate the probability that the word occurs in that set. In other words, we calculate, $P(w|R_q)$ and we use this to approximate $P(w|M_r)$:

$$P(w|M_r) \approx P(w|R_q) = \sum_{d \in R_q} P(w|d)P(d|q) \tag{3}$$

We note that computing Equation 2 over all the documents in the dataset is very expensive and unnecessary. Since the number of documents is very large, the value of $p(d|q)$ is quite close to zero which guarantees the sum of over all documents is one. *Therefore, unlike [5], in our work we only select documents with the highest $p(d|q)$ while computing Equation 2.* We expect this to both save execution time and help reduce the effects of smoothing.

3. SNIPPET EXTRACTION

In this section, we present curvature analysis driven snippet extraction approach based on the language model. In [10], Qi and Candan presented a curvature analysis based approach to text segmentation. In this approach, windows of text (referred to as entries) from a given document are each represented as TF/IDF-based keyword vectors and dissimilarity between pairs of entries are computed. The entries are then mapped into a 1D space by applying multidimensional scaling [10] on the resulting dissimilarity matrix.

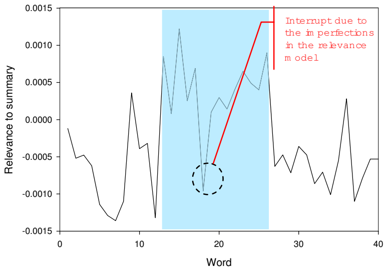

Figure 1. Curvature analysis: the x-axis represents the sequence number of words, while the y-axis represents the relevance of each word to the snippet, based on the language model. The shaded area marks the boundaries of the snippet in the document

The 1D space is extended with the entry sequence number to obtain a *topic development curve*, which represent how topic develops in the given text, is created and this curve is then segmented into components, based on slope (i.e., rate of topic change), concentration (i.e., focus), and interruption analysis.

In our work, query relevant segmentation is used for differentiating the part of the document relevant to the query from the parts of the document that are irrelevant. For this purpose, we interpret the probability, $p(w|M_r)$, as the similarity degree between the word, w, and the query. Similarly, $P(w|M_{\bar{r}})$, is the probability of observing a word, w, in a document that is irrelevant to the query-relevant snippet. Based on these two, we can map a given document into a curve in 2-dimensional space, where the x-axis denotes the sequence number of the words in the document, while the y-axis denotes the degree with which a given word is related with the snippet. This degree is calculated by subtracting $P(w|M_{\bar{r}})$ from $p(w|M_r)$. Thus, the mapping is such that the distances between points in the space represent the dissimilarities between the corresponding words, relative to the given query.

As shown in Figure 1, the resulting relevance-curve reveals the relationship of the text content of the document with the required snippet. In this case, finding the most relevant snippet is converted to the problem of finding a curve segment, where the sum of points in y-value is the largest: that is, most of words in that curve are generated by the relevance model M_r.

Note that the curves generated through the language models are usually distorted by various local minima (or *interrupts*). For example, due to the approximate nature of the relevance model, M_r, some relevant words within the snippet may not be correctly modeled. This may, then, result in local minima in the curve. Such an interrupt is highlighted in Figure 1. While, the topic development curve in [10] is generated differently and represents inter-entry difference as opposed to the estimated relevance of a given word to the snippet being searched, we can still leverage the boundary detection approach presented in [10] to prevent such interrupts from resulting in over-segmentations.

Table 1. Dataset used for evaluation

No. of queries	30
No. of documents	987
Average doc length	457 words
Average snippet length	193 words

Table 2. Performance of alternative methods

Method	Precision	Recall	F-measure	Time cost (ms/doc)
FS-Win	0.76	0.63	0.69	18.6
RM-Curve	0.84	0.87	0.85	40.1

Thus, we first segment the given relevance-curve using the curvature analysis scheme presented in [10]. After obtaining the segments, we compute the total relevance (i.e., sum of the relevance values for each word) for each segment. We then identify and present those with the largest relevance totals as the query-relevant snippets.

4. EXPERIMENTALS

To gauge how well our proposed snippet identification approach performs, we used a question-answer collection collected from the Internet. The collection contained 1600 question/answer pairs in total. We then selected 30 popular questions and extracted noun words from these questions to use as queries. The corresponding answers of these 30 questions are regarded as the ground truth (relevant snippets). The relevant and irrelevant snippets are then randomly merged into a collection of about 1000 documents. Note that in this synthetic dataset, entries have well defined snippet boundaries. We will also report results with noisy data. A detailed description of dataset is shown in Table 1.

As introduced before, curvature analysis (RM-Curve) is applied to extract variable-length snippets under the framework of Relevance Model. In addition to the curvature analysis method to extract relevant snippets, we also implemented a baseline method (FS-Win). For this purpose, we used the popular window method, commonly applied to present snippets in the search-result pages [6]. Given a window size k, we examine all k word long text segments and select the one which has the most occurrences of the query words to be the most relevant. In our experiments, the window size k is set to about 200 words, the average word length of the ground truth snippets for the dataset (Table 1).

Given the identified snippets, we computed precision, recall and F-measure values as follows: Let L_s be the length of the ground truth snippet. Let L_e be the word length of the

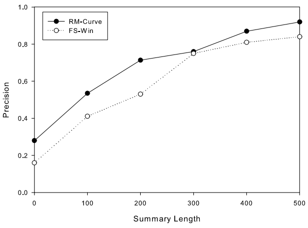

Figure 2. Effect of length on precision

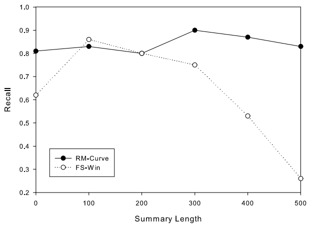

Figure 3. Effect of length on recall

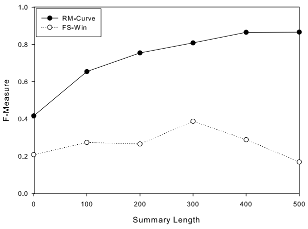

Figure 4. Documents with Noise Information

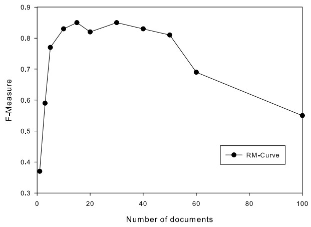

Figure 5. Number of document used in modeling

extracted snippet and L_o be the overlap between the ground truth and the extracted snippet. Then,

$$\mathrm{Pre} = \frac{\mathrm{L_o}}{\mathrm{L_e}}, \mathrm{Rec} = \frac{\mathrm{L_o}}{\mathrm{L_s}}, \mathrm{F} = \frac{2\mathrm{Pre} \bullet \mathrm{Rec}}{\mathrm{Prec} + \mathrm{Rec}}$$

We can see from Table 2 RM-Curve outperforms FS-Win with remarkably high (0.84-0.9) precision and recall. Further analysis of the results showed that, in general, RM-Curve extracted larger snippets, which correctly included the ground-truth snippets to achieve high recall, but reducing the precision. In terms of time cost, both algorithms perform very fast. Even though the baseline is the fastest, the precision and recall gains of RM-Curve makes it desirable despite its slightly higher computation time.

As shown in Figure 2, predictably, the precision improved for both schemes as the size of the relevant portion of the text increased. However, in Figure 3, we see that the recall for the FS-Win can achieve a high recall only when the window length matches the length of the actual snippet almost perfectly. On the other hand, RM-Curve has highly stable recall rates independent of the length of the ground truth. As a conclusion, we can state that RM-Curve shows a good performance with identifying variable-length snippets.

So far, we experimented with data with clean delineation of relevant and irrelevant parts. To observe the affects of more realistic situations, we inserted query words into two different text content areas of the documents. Thus, there are three text segments in the relevant documents containing the query words, only one corresponding to the ground truth; the other two are noise. As shown in Figure 4, RM-Curve performs almost the same as the without noise situations in Figure 2 and Figure 3. However, the performance of FS-Win dropped significantly. In other words, RM-Curve is able to select query-relevant snippets not only considering query words but also the contextual words related with the query. Thus, when there is noise in the document, it can filter out irrelevant segments which have weak connections with these query words.

We also explored the effect of the size of retrieved relevant document set, R_q, in Equation 2. As shown in Figure 5, the performance is stable when the number, n, of the relevant document set is set to anywhere between 5 to 40 documents. Using fewer than 5 or more than 40 documents has negative effects. In the experiments reported above, we used 15 top-ranked documents in estimating the relevance model

5. Conclusions

In this work we have shown how the relevance model technique can be extended to query-relevant snippet identification task. Our proposed methods, RM-Curve is able to locate the exact place of the most relevant, query-oriented snippet. We have demonstrated a substantial performance improvement using the relevance model base scheme. The critical parameter selection problem was also shown to be relatively simpler than other language model based approaches, since our approach of selecting $top-n$ documents for building the relevance model help prevent the performance to be very sensitive to the smoothing parameters.

References

[1] J. Callan. Passage-level evidence in document retrieval. *ACM SIGIR*, 1994.

[2] Google. *www.google.com*.

[3] M. Hearst. Texttiling: A quantitative approach to discourse segmentation. *Computational Linguistics*, 1997.

[4] M. Hearst and C. Plaunt. Subtopic structuring for full-length document access. *ACM SIGIR*, 1993.

[5] V. Lavrenko and W. Croft. Relevance-based language models. *ACM SIGIR*, 2001.

[6] R. Nobles. Pleased with your google description? *Etips*, 2004.

[7] OASIS. Organizing, annotating, and serving information to individuals without sight. *http://icare.eas.asu.edu/oasis/*.

[8] J. Ponte and W. Croft. Text segmentation by topic. *ECDL*, 1997.

[9] J. Ponte and W. Croft. A language modeling approach to information retrieval. *ACM SIGIR*, 1998.

[10] Y. Qi and K. Candan. Cuts: Curvature-based development pattern analysis and segmentation for blogs and other text streams. *HYPERTEXT*, 2006.

[11] R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? *Proceedings of the IEEE*, 2000.