

# A Language Modelling Approach to Relevance Profiling for Document Browsing

David J Harper, Sara Coulthard and Sun Yixing

The Robert Gordon University  
School of Computing  
St. Andrew Street  
Aberdeen AB25 1HG  
United Kingdom  
+44 1224 262706  
djh@scms.rgu.ac.uk

## ABSTRACT

This paper describes a novel tool, SmartSkim, for content-based browsing or skimming of documents. The tool integrates concepts from passage retrieval and from interfaces, such as TileBars, which provide a compact overview of query term hits within a document. We base our tool on the concept of relevance profiling, in which a plot of retrieval status values at each word position of a document is generated. A major contribution of this paper is applying language modelling to the task of relevance profiling. We describe in detail the design of the SmartSkim tool, and provide a critique of the design. Possible applications of the tool are described, and we consider how an operational version of SmartSkim might be designed.

## Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – *retrieval models, search*. H.3.7 [Information Storage and Retrieval]: Digital Libraries – *user issues*. H.5.2 [Information Interfaces and Presentation]: User Interfaces – *evaluation/methodology, graphical user interfaces (GUI)*.

**General Terms:** Design, Human Factors, Theory.

## Keywords

Information retrieval, language modeling, browsing and reading appliances, e-books, user interfaces, visualization.

## 1. INTRODUCTION

In step with the increasing volume of text in digital form, we have observed a corresponding increase in the size of individual digital documents. Governments are making available large number of public records in digital form, e.g. parliamentary proceedings, government reports, patents, etc, which by their nature tend to be

lengthy. Companies are delivering company reports, product information and product manuals in digital formats. Electronic books contain large chunks of text, even allowing that books are often organised into sub-units, such as chapters. Given these large text documents, a digital library user is faced with the daunting task of identifying relevant material within a document. Thus, in addition to the tasks of collection selection (or equivalently search engine selection on the Web), and document retrieval, the user is faced with the additional task of within-document retrieval. Various approaches have been proposed for within-document retrieval, including passage retrieval and user interfaces supporting content-based document browsing, e.g. TileBars[1]. In this paper, we describe the SmartSkim tool, which integrates passage retrieval and content-based document browsing. SmartSkim is based on the concept of relevance profiling, in which a profile of retrieval status values across a document is generated. We show how language modelling can be used for relevance profiling.

In document retrieval, an overview of the retrieved documents is typically provided in the form of a ranked list of document surrogates. This overview enables a user to identify potentially relevant documents, and to select and open these document for display. Our purpose is to provide an appropriate overview for within-document retrieval. Such an overview should enable a user to identify relevant passages of a document in respect of a query, and allow the user to select and browse these passages. It is intended that this within-document retrieval overview will be an integral part of a document reading/browsing tool. Relevant passages will be displayed in the context of the complete document, and within-document navigation and browsing will be seamlessly integrated.

The SmartSkim overview is based on a relevance profile, displayed as a histogram (see Figure 1). Based on a query input by the user, a relevance profile is computed over the document. Each bar of the histogram corresponds to a fixed length section (tile) in the text of the document, and the height of the bar corresponds to the computed relevance of that tile. By clicking on a bar, the corresponding tile within the document is centred in the document viewer, and query term instances are highlighted. Thus, potentially relevant passages can be identified, navigated to by clicking on a histogram bar, and browsed in the context of the complete document. It is clear from the displayed profile, that there are two main sections of the document relevant to “Cooper’s expected search (length)”.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

JCDL'02, July 13-17, 2002, Portland, Oregon, USA.

Copyright 2002 ACM 1-58113-513-0/02/0007...\$5.00.

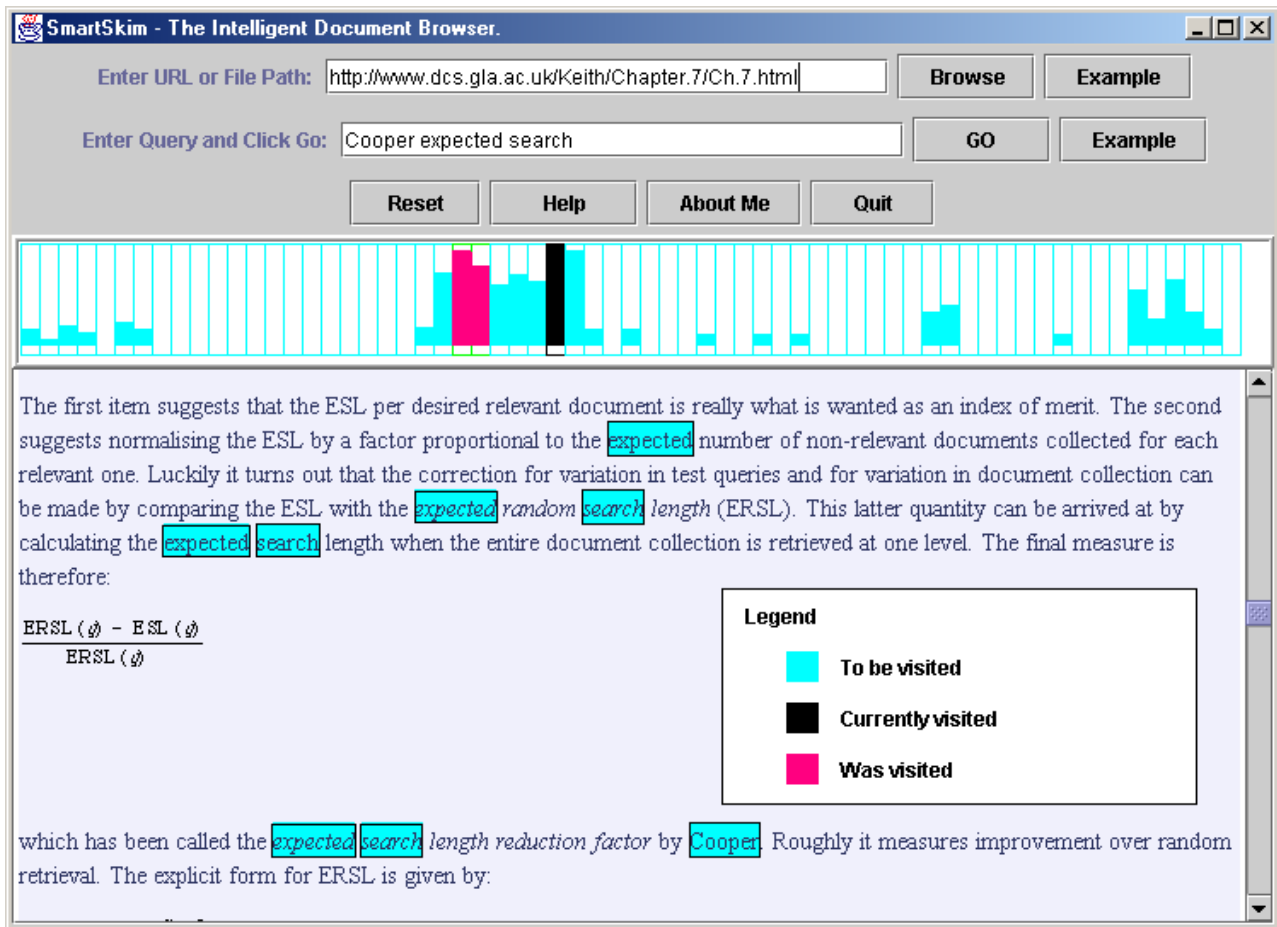


Figure 1: Screenshot of SmartSkim interface

The first coincides with a lengthy treatment of this subject halfway through Chapter 7 of van Rijsbergen's classic textbook "Information Retrieval", and the second with a section on further reading.

The SmartSkim overview should enable a user to:

- Make judgements on the overall relevance of a document, and its coverage with respect to the query; and
- Identify relevant passages within the document, and thus support information extraction and fact finding.

There are similarities between our concept, and that of TileBars [1] and the SCAN system [2]. TileBars provides an overview for within-document retrieval, based on visualizing query term occurrence patterns within text tiles (document passages). SCAN provides a relevance profile over speech documents for within-speech retrieval. Importantly, we base relevance profiling on a solid theoretical foundation provided by language modelling.

The structure of this paper is as follows. We review the main approaches to within-document retrieval, and argue that relevance profiling combines the best features of passage retrieval, content-based browsing and retrieval overview. Then, we describe how language modelling can be used to generate a relevance profile given a document and query. The design of the SmartSkim tool is

described in detail, in order to emphasize that the minutiae of user interface design can potentially have large impact on usability. And, the design is also critiqued. We describe possible ways in which relevance profiling could be used within operational systems. Then, possible architectures for an operational SmartSkim system are outlined. Finally, we provide some concluding remarks and identify future work

## 2. RELATED WORK AND BACKGROUND

In reviewing relevant work, we concentrate of within-document retrieval, and specifically the following issues: relevant passage retrieval; user interface support for relevant passage identification, selection and browsing; and integration of within-document retrieval and document browsing. Summarisation and query-biased summarization are also briefly reviewed.

Much of the research on passage retrieval has concentrated on segmentation of documents into passages, retrieval models of passage retrieval, and efficient implementations of the models [3] [4] [5]. Moreover, often passage retrieval is viewed as a means of improving document retrieval, rather than as a means of specifically retrieving passages [4] [5]. Although most authors acknowledge that it would be beneficial to retrieve and display passages for end users,

this is, by and large, not the main focus of their work. Hearst developed a process called text tiling (passage identification), which is the basis for the TileBars interface (see below for a description).

Some work on within-document retrieval is concerned with visualizations that show the relationships between query terms, and their occurrences within the document, or passages thereof. A simple and effective form of visualization is highlighting query terms in the displayed document, and enabling the user to scroll between query term occurrences [6] [7]. However, the relevance of any given passage to a query must be established after scrolling to the passage, and passages are visited in within-document order rather than in order of relevance. Recent work by Byrd has extended highlighting using colour coding of each query term (as have others), and more importantly integrating a color-coded overview within the document scrollbars [8]. However, the user must determine the relevance of a passage using this information.

TileBars provides a compact overview of the query term occurrences (actually query term facets) within each text tile (consecutive non-overlapping sequences of words) [1]. The document overview contains a vertical bar per tile, and within each bar the density of occurrences of each query term is shown using grey-scale shading. The user must judge the likely relevance of a tile (passage) using this information, and by clicking on a bar, centre the document viewer on the corresponding tile within the document. Arguably, asking the user to assess relevance in this way adds to the cognitive load of the user, possibly at the expense of browsing relevant parts of the document.

The SCAN system provides an overview facility for within-document retrieval for speech documents [2]. The overview takes the form of a histogram display, in which each bar corresponds to an audio paragraph (paratone), and the height of the bar indicates the overall query term weights for that paratone. Colour coding within each bar provides additional feedback on which query terms occur in the paratone. This overview allows relevant audio paragraphs to be identified, and by clicking a bar, playing back the corresponding speech passage. SCAN therefore provides many of the functions we consider important for within-document retrieval, albeit in the context of spoken document retrieval.

In [9], locality-based similarity is introduced, in which a relevance profile is computed across the document at each word position. The score at each word position is determined by the contribution of query term occurrences within the locality of the given word position. The authors do not specifically address how this profile would be used in a within-document retrieval tool, but that is clearly the intention.

Document summarization refers to a condensation of a full-text document as a summary or abstract, where the resultant summary presents succinctly the objectives, scope and findings of a document. In general, such summaries are generic, and thus not adapted to a specific information need. In [11], the idea of query-biased summarization is introduced, in which summaries are generated relative to a query. This work is mainly aimed at providing good document surrogates, for use in displaying the results returned by a search engine, i.e. typically a list of ranked document surrogates. However, after determining whether a document is relevant, a user must typically browse the document to extract the information required. It is this browsing process with which we are mainly concerned.

The design of the SmartSkim tool was inspired in part by the TileBars system, and has much in common with the SCAN system. SmartSkim provides a graphical overview of the document relevance profile, as does SCAN. Direct access to relevant passages is provided via an interactive overview, as happens in both TileBars and SCAN. We purposefully have not provided feedback on query term occurrences, believing that users should be encouraged to rapidly browse potential relevant passages rather than devoting cognitive bandwidth to interpreting the overview. Moreover, we believe that the language modelling basis for the relevance profiling will provide a good indication of passage relevance, judging by the performance of language modelling when used in document retrieval [11].

### 3. RELEVANCE PROFILING USING LANGUAGE MODELLING

The goal of this work is to enable content-based browsing of documents. Given a query, we want to compute a relevance profile across the document, in which effectively a retrieval status value (RSV) is computed for each word position in the document. This RSV will be based on a *text window*<sup>1</sup> (fixed number of consecutive words) associated with each word position. We use language modelling to construct a statistical model of a text window, and based on the model, we compute the probability of generating a query. This computed probability is the retrieval status value for the text window. In the following, we describe our novel application of language modelling to relevance profiling.

A language model is a statistical model of text, which captures the statistical distribution of text features. Thus, in principle we can model any piece of text using language modelling, and in this application we build models over text windows. We adapt a simple word-based model that was used for document retrieval [1], together with refinements described in Song and Croft [12].

In the Ponte and Croft model, a document is modelled by a probability distribution over the terms of the vocabulary,  $P(\text{term} / \text{document})$ . The probability that a document is relevant to a query is equated with the probability of generating the query, given the model of the document:

$$P(\text{query} | \text{document}) = \prod_{t_i \in \text{query}} p(t_i | \text{doc}) \prod_{t_i \notin \text{query}} (1 - p(t_i | \text{doc})) \quad (1)$$

In Song and Croft, only query term occurrence is considered, and the second term in (1) is dropped. In addition, the simple term model for a document is replaced by a mixture model, which mixes the document (doc) and collection (coll) term distributions, in the proportions given by  $w_{\text{doc}}$  and  $(1 - w_{\text{doc}})$ , where  $w_{\text{doc}}$  takes a value between zero and one inclusive. This is done to smooth probability estimates that would otherwise be derived from small samples of text, that is, only the document.

---

<sup>1</sup> We use the term “window” in that a sliding text window moves across the document in order to compute the RSV at each word position in the document.

As a result, (1) is replaced by:

$$P(\text{query} | \text{document}) = \prod_{t_i \in \text{query}} p_{\text{mis}}(t_i | \text{doc}) \quad (2)$$

where

$$p_{\text{mis}}(t_i | \text{doc}) = w_{\text{doc}} * p_{\text{doc}}(t_i | \text{doc}) + (1 - w_{\text{doc}}) * p_{\text{coll}}(t_i | \text{coll})$$

In the case of relevance profiling, we wish to estimate the probability of generating a query, given parts of the document, namely text windows. It is worth noting that one of the strengths of language modelling is that it can be applied to any piece of text, however we wish to define that. In a sense, it frees us from thinking about documents (or passages) and document (or passage) retrieval, and we can simply model the most appropriate pieces of text for the given application. However, it is the case that language models will be less accurate when used to model smaller pieces of text, and this may potentially impact on the accuracy of relevance profiling.

For relevance profiling, the application of the Song and Croft model is quite straightforward. We model the distribution of terms (actually words) over a text window, as a mixture of the text window and document term distributions (analogous to the use of document and collection in the Song and Croft model) as follows:

$$P(\text{query} | \text{window}) = \prod_{t_i \in \text{query}} p_{\text{mis}}(t_i | \text{win}) \quad (3)$$

where

$$p_{\text{mis}}(t_i | \text{win}) = w_{\text{win}} * p_{\text{win}}(t_i | \text{win}) + (1 - w_{\text{win}}) * p_{\text{doc}}(t_i | \text{doc})$$

Thus, the probability of generating words is determined in part by the text window, and in part by the document in which the window is located. The estimates are smoothed by the document word statistics using the mixing parameter,  $w_{\text{win}}$ <sup>2</sup>. Importantly, it means that if a particular query term does not appear in a text window, but does appear in the document, then we can estimate a non-zero estimate for  $P(\text{query} | \text{window})$ . Note, however, if *none* of the query terms appear in a given window, then by definition, the probability of generating a query is set to zero.

The individual word probabilities are estimated in the obvious way using maximum likelihood estimators:

$$p_{\text{win}}(t_i | \text{win}) = n_{iW} / n_W \quad (4)$$

$$p_{\text{doc}}(t_i | \text{doc}) = n_{iD} / n_D$$

where  $n_{iW}$  ( $n_{iD}$ ), and  $n_W$  ( $n_D$ ), are the number of word occurrences of word  $i$  in the window (document), and total word occurrences in the window (document) respectively.

The relevance profile is given by the retrieval status value at each word position  $i$ :

$$RSV_{\text{window}}(i) = P(\text{query} | \text{window}_i) \quad (5)$$

where text window  $i$  is the sequence of words  $[w_i..w_i+L_W-1]$ , and  $L_W$  is the fixed length of each text window. Note, that notionally the document is extended by additional  $L_W$  words (stopwords) to allow the relevance profile to be computed across the entire document.

<sup>2</sup> The best value for this parameter needs to be determined empirically, and we have used 0.8 in our system.

In order to provide a plot of the relevance profile, and to support direct navigation to relevant parts of a document, we will argue in the next section that the retrieval status values should be aggregated over fixed size, non-overlapping sections of text we call *text tiles*. Here, we provide the basis for aggregating RSVs. We assume that the document text is divided into fixed length, non-overlapping text tiles. Let us assume that each tile is  $L_T$  words long. The aggregate RSV for a given tile  $j$  is given by:

$$RSV_{\text{tile}}(j) = \text{agg-fun} \quad (6)$$

$$(\{RSV_{\text{window}}(i), i = (j-1)*L_T + 1 .. j*L_T\})$$

Examples of aggregate functions (agg-fun) include average, minimum and maximum, and we opt for the maximum for reasons presented in the next section. Note, that each of the windows, which potentially contributes to the RSV for the text tile, starts within the tile, but may extend beyond the end of the tile.

Text windows and text tiles, although related, serve two different purposes. A text window is used to compute an RSV at each word position in the document. The fixed size of a text window is set to the “typical” size of a meaningful chunk of text, such as the average size of a paragraph (or possibly section). The average size of a paragraph can be determined empirically, and in our system we have set it to 200. A text tile is used to aggregate or combine the RSVs of *all* text windows that *start* within the given tile, and tiles are used for summarizing (and thence displaying) relevance profiles. The size of a tile is fixed for a given document, and we use a minimum size of 200 in our system. For large documents, the tile size may need to be increased (see next section).

In the next section, we show how the relevance profile can be presented to a user through an interface component, namely a relevance profile meter, that allows relevant parts of the document to be identified, visited and browsed.

#### 4. DESIGN OF THE SMARTSKIM USER INTERFACE

The design of the SmartSkim user interface is presented, together with the supporting rationale for that design. Before that, we provide an overview of the SmartSkim tool, in order to make more concrete our discussion of the design.

Currently, SmartSkim is designed as a standalone tool. The main function of the SmartSkim tool is to support a user in rapidly browsing (or scanning) a document in order to locate relevant parts of a document given a query. Figure 1 shows a snapshot of the interface. A user opens a document to skim, inputs a query, and a relevance profile meter displays the relevance profile across the document for that query. The document itself is displayed in the document viewer, and all query words (and variants thereof) are highlighted. The relevance profile meter is central to the operation of SmartSkim, and will be the main focus of our design discussion. The vertical bars on the meter correspond to sections of the document, referred to as tiles, and the height of each bar corresponds to the summary retrieval status value of the corresponding tile. We discuss the rationale for “tiles” below, and the method for mapping the relevance profile (see previous section) to the meter. When the user clicks on an individual bar, then the document display scrolls to the appropriate section (tile) in the document. Effectively, the bars of the relevance profile meter act as “hot links” into the body of the document.

In the following, we will describe and critique the design of the following components of the interface:

- Relevance profile meter (RPM)
- Document Viewer
- Query Input

Then, we will critique the overall design using Green’s Cognitive Dimensions Framework [13].

#### 4.1 Relevance Profile Meter

The relevance profile meter displays a graphical representation of the information in the relevance profile. The intention is to enable a user to identify relevant sections of the document, and by interacting with the meter move directly to that section of the document. In mapping the relevance profile to some kind of meter, the main decision we faced was how to display the relevance status values at each word position in the document. We could display the values for each word position as a graph plot, and then interact with the plot to move to the corresponding word in the document. Alternatively, we could use an aggregate-type plot (e.g. histogram), which displays an averaged RSV over some range of word positions as a bar. The advantages of the second option is that we could use the “bar” both as a focus for interaction, and as means of providing direct feedback about the sections of document that had been visited. Moreover, we could describe the bars as “hot links” to potential users, and leverage the existing user models of familiar systems, i.e. web browsers. We believe that the bar/section, rather than graph point/word position will be simpler to present and reinforce through the interface.

Having decided in a favour of the aggregate-type plot, we considered just one way of displaying the information. The obvious choice was a straightforward histogram plot, and this was adopted because of its familiarity. The plot could be displayed horizontally across the screen, or vertically down the screen. The horizontal plot is familiar as the vertical corresponds to the dependent variable, i.e. the RSV for each document section. The disadvantage is that we can only fit a relatively small number of bars across the meter, which means that for very large documents we may need to dynamically adjust the size of the sections (text tiles). On the other hand, a vertical plot can be adjusted to the vertical size of the document display, and indeed the RSV bars can be integrated as part of the document scrollbars; see [8] for a similar use of scrollbars. A vertical scrollbar corresponds to the entire document, and thus provides a “natural” place for displaying a relevance profile meter. However, there is still a problem of adjusting the number of bars to the vertical display. Also, the scrollbars may need to be increased in width to accommodate the RSV bars. On balance, we opted for the horizontal display because of user familiarity with it, and because of ease of implementation. We do not provide horizontal scrolling within the relevance profile meter. We wanted the user to be able see all of the relevance profile on the display, and aid rapid selection of the most relevant text tile(s) in which to begin browsing.

In the previous section, we showed how the relevance profile of retrieval status values at each word position could be aggregated to produce aggregate RSVs for a given text tile. We have chosen to use the maximum aggregate function, as this corresponds to using the largest RSV of any window beginning within the given text tile as being representative of that tile.

The height of the bar in the relevance profile meter is scaled to the RSV of the corresponding text tile. This appears to be a straightforward matter of scaling the bar heights. Crucially, however, a user will rely on the heights of the bar to assess both absolute relevance and comparative relevance of tiles.

Let us consider comparative relevance between bars. RSVs are probabilities, and they take values between zero and one. However, because of the way probabilities are estimated, these probabilities are likely to be very small. Therefore, they should be scaled relative to the largest probability computed. But, are probabilities best for displaying comparative relevance? Suppose that we have a query of  $qt$  terms, and suppose in given tile A, the best window A1 contains a single occurrence of each query term, and for tile B, the best window B2 contains two occurrences of each query term. What is the comparative height of the bars if probabilities are used? Taking eqn (3) and supposing for this example  $w_{win}$  is set to 1, then:

$$\begin{aligned} P(\text{query} / \text{window A1}) &= (1/N_w)^{qt} \\ P(\text{query} / \text{window B2}) &= (2/N_w)^{qt} \\ &= P(\text{query} / \text{window A1}) * 2^{qt} \end{aligned}$$

Hence, the bar for tile B would be  $2^{qt}$  higher than that of bar for tile A. We would argue that this might not accord with a user’s intuition. That is, twice as many query term occurrences result in a much more than twice the height of bar. Suppose, we use of log scale for the probabilities, then the values on which the bar heights would be based are:

$$\begin{aligned} \log_2 P(\text{query} / \text{window A1}) &= qt \log_2 (1/N_w) \\ \log_2 P(\text{query} / \text{window B2}) &= qt \log_2 (1/N_w) + qt \\ &= \log_2 P(\text{query} / \text{window A1}) + qt \end{aligned}$$

Thus, doubling the number of query terms, results in a linear increment in the height of the bar. We would argue that logarithmic scaling would make it easier to compare relevance values across bars. We used logarithmic scaling in our prototype.

Let us now consider using the height of a bar as an absolute indication of relevance. How can we add scale marks to indicate absolute degree of relevance? Clearly, probabilities lie between zero and one, and on a log scale between minus infinity and zero. However, sensible minimum and maximum values on the scale are needed to ensure bars are useful for both comparative and absolute judgements of relevance. We opted to use the actual *minimum* and *maximum* values attained for  $\log P(\text{query}/\text{window})$ , and scale between *minimum*  $-1$  and *maximum*  $+1$  with the following proviso. It was thought useful to include and have visible scale marks corresponding to following average numbers of query term occurrences per window: 0.5, 1, 2, and  $4^3$ . This proviso requires that in some cases the maximum and minimum need to be adjusted. In our initial prototype, we did not include scale marks.

<sup>3</sup> For example, in the sample skim shown in Figure 1, the number of query term occurrences in the sliding window (fixed at 200 words) was one, five and five respectively for the query terms as shown. This is certainly quite a high number, and a consequence of the detailed treatment of “Cooper’s expected search length” in that part of the document.

To assist the user in browsing the document using the relevance profile meter, and the bars thereof, we thought it important to provide feedback as to which bars (and corresponding tiles) had been visited. Therefore, we used colour coding to indicate which bar/tile had: yet to be visited (cyan), currently being visited (magenta) and visited (green). This colour-coding scheme corresponds broadly to those used typically in web browsers.

## 4.2 Document Viewer

The document viewer displays a document, and initially the beginning of the document is displayed. Documents of MIME-compliant types html, RTF and text can be displayed. All words, which stem to the query term stems, are highlighted to assist the user in locating relevant text. We chose to highlight in cyan, to reinforce the fact that initially cyan bars on the relevance profile meter correspond to cyan-highlighted (query) words in the document.

A user may browse the document using a combination of direct navigation via bars of the relevance profile meter, or by scrolling the document. When the user clicks on a bar within the RPM, the document viewer positions the display at the start of the corresponding tile.

In designing the viewer, we considered providing feedback in the *displayed document* about which parts of the document had been visited/viewed. We contend that it would be too difficult to maintain a consistent view of “visited” and “viewed”, given that both direct navigation and scrolling were supported. However, we re-visit this decision in the design critique below.

## 4.3 Query Input

Queries are subject to the same the text analysis as the documents. As a result of query analysis, text within the query is labelled as: non-word text, stopwords, words that do not appear in the document, and words that do appear in the document. This information should be fed back to the user, in order to reinforce user’s mental model of relevance profiling. We propose colour-coding the query text as follows. Non-word and stopword text is coloured black, as these “words” are ignored in relevance profiling. Words that appear in the document are colour-coded cyan, and this colour is also used in highlighting these same words when the document is displayed. Words that do not appear in the document are colour-coded red, as potentially these words may either by misspellings or they are actual words that do not appear. In either case, the user should understand that these “words” do not contribute to the relevance profile. This query feedback will be implemented in a subsequent version of SmartSkim.

## 4.4 Critique of Design

We have described the design of the SmartSkim interface in detail, as we believe that it is all too easy to make interface design decisions without due consideration of the consequences. We hope this detailed discussion has emphasised the need for purposeful design, even for apparently trivial aspects of an interface.

The Cognitive Dimensions Framework can be used to map out the design space and/or understand the tradeoffs in designing information seeking environments [13]. These dimensions have been used to assess the strengths and weakness of a variety of cognitive artifacts, e.g. spreadsheets [14]. We will use (some of) these dimensions in discussing the design of SmartSkim.

**Closeness of mapping** This dimension is concerned with the extent to which an artifact is designed for, and related to, its purpose or

function. SmartSkim is designed for content skimming of text documents, and we strove to include only functions related to that purpose. In part, we rejected feedback on individual query terms in the relevance profile meter as we considered it might damage the smooth transition between identifying relevant passages and deciding to browse them.

**Role Expressiveness** This dimension refers to the extent to which the relationship between objects we compute with is made manifest in the interface. We purposefully designed the relevance profile meter so the relationship between a bar in the meter, and a text tile was reinforced (although we could probably go further). Again, the relationship between query terms in the query box, and those highlighted in the document display is reinforced using colour. Feedback is (will be) provided to the user concerning query words that contribute to the relevance profile.

Role expressiveness is highly important in creating an accurate mental model of a system by the user. As designed, the relevance profile meter is not updated when text tiles are “visited” during scrolling within the document viewer. However, the relationship between the RPM and the visited tiles in the text is clearly very important, and should be maintained and reinforced. This deficiency is the current design will be addressed in the next version of the interface.

**Secondary Notation** This dimension refers to typographic and layout cues that lie outside the formal computation/notation of a system. Thus, for example, the extensive use of colour coding within the interface provides useful secondary notation. However, the user has little control over the way their work is arranged, managed and annotated on the display. We would surmise that annotation of tiles would be a very useful feature, as indeed would be the ability to manage multiple canned queries.

**Side-by-side-ability** This dimension refers to the extent to which information objects can be compared side by side, as this is known to be an important function in information seeking. SmartSkim enables relevant passages to be browsed in the context of the full document, and thus side-by-side with contiguous passages. However, SmartSkim does not provide support for side-by-side comparison of non-contiguous passages, which might arguably be useful, e.g. comparing an introduction against the conclusions.

This critique of the interface uses just some of the Cognitive Dimensions, and yet we hope we have justified in part some of the design decisions we made when designing SmartSkim.

# 5. APPLICATIONS AND ARCHITECTURE OF SMARTSKIM

## 5.1 Applications

There are a number of possible applications of the SmartSkim tool, and of the underpinning relevance profiling concept.

Most obviously, SmartSkim could be integrated within an information retrieval (IR) system, and provide a within-document retrieval capability to supplement the usual document retrieval capability. Parts of the SmartSkim functionality could be used in two ways. First, relevance profiles could be displayed with document surrogates as part of the ranked list view of the retrieved documents. Clearly, the query used for retrieving the documents would be used for generating the relevance profile for each document. The relevance profiles would provide additional information for the user to assess the relevance of a document, prior

to asking for the full document to be displayed. Arguably, additional information about the patterns of occurrence of individual query terms for each tile (as in TileBars) would also prove useful in this setting. Second, the entire SmartSkim tool could effectively replace the document viewer component of an IR system. On displaying the document, a relevance profile based on the original query could be used. Subsequently, other within-document queries could be used to explore a document from different perspectives. For example, a general query may be posed on “government regulation of sub-sea submersibles in oil pipeline inspection”. A (large) retrieved document on this topic might be explored by the user from the perspectives of “visual inspection”, “fail-safe remote control” and “non-destructive testing of pipelines” (say).

SmartSkim could be integrated with a document reading information appliance[15], such as XLibris [16], to add content-based document browsing. Hence, e-publications such as books, newspapers, magazines, etc. could be content-browsed using SmartSkim. O’Day, studying professional searchers, has identified three types of typical search activity [17]. There are following an information-gathering plan specific to the task at hand; monitoring a well-known topic over time; and exploring a topic in an undirected fashion. We will briefly consider these three activities in the context of within-document retrieval using SmartSkim. For information gathering, query-based browsing would be invaluable for intensively studying texts. This includes finding co-located characters in novels, exploring the use of given phrases and idioms, and for use as an adjunct to the back-of-book index. For example, a student might be asked to investigate the role of social activities in the novel “Lord of the Rings”, and hence content-browse the corresponding e-book using appropriate queries such as “eating, drinking, meal, feast, banquet, table”, “singing, dancing, songs, verses, music, instruments”, etc. For monitoring a well-know topic, one could imagine a user setting up a number of “canned” queries to aid him/her in purposeful scanning of e-publications. SmartSkim would have to be extended minimally to support canned queries. Thus, a financial analyst might set up queries to enable her/him to scan electronic publications for information of various aspects relating to the financial markets. Such publications might include purchased e-newspapers, e-periodicals borrowed from a digital library, and so on. For undirected exploration of a document, a user could issue very broad queries to SmartSkim. However, the tool is less suited to this activity, and was not specifically designed for undirected exploration of a topic.

Currently, a document is subdivided into fixed length text tiles for the purposes of relevance profiling. However, our language modelling approach can be applied to any arbitrary chunk of text with a document. A document may indeed be already sub-divided into logical sections in some way, and this is particularly so for web documents, which may be marked-up using HTML or XML. Using our approach, we can compute a retrieval status values (RSV) as per equation (3), for each *logical subdivision*, and present the resultant RSVs directly using a relevance profile meter.

The concept of profiling a document can be used for any measure we can compute across the text of a document. For example, we could compute readability scores across a given text, and present the results in a readability profile meter. This could be used by an editor/author to highlight and thus improve difficult-to-read parts of a document. For other digital media, we could envisage various useful profile instruments, e.g. highlighting the (likely) points at which a particular kind of instruments is being playing in an audio

recording; identifying scenes in a movie that are (likely to be) action scenes, etc. These will depend on being able to compute an appropriate measure, e.g. action-ness of a scene.

The main focus of this paper is, however, text documents, and we have presented a technique for relevance profiling text documents, and a tool for browsing text documents based on these profiles.

## 5.2 Architecture

The SmartSkim system comprises two main modules, a profiling “engine” and the user interface, and it is implemented in Java. The SmartSkim profiling “engine” uses an inverted file constructed for a document. For each occurrence of a word, we record the corresponding word stem, the word position in the text, the character position and the word length. The word position is used to compute the relevance profile. The character position and word length are primarily used for highlighting all word variants corresponding to the stemmed query words. The SmartSkim user interface is implemented using standard Java Swing components.

The current SmartSkim prototype is a standalone system, which is appropriate for a research prototype. Internally, the system is organised as “client” module and “server” modules, which communicate via methods calls. However, the system has not been actually divided into separate client and server programs. There are some limitations in the current system. Currently, every time a document is opened, we re-analyze the document, which is highly inefficient. However, the use of inverted files does enable a relevance profile to be computed very efficiently.

For a production version of SmartSkim, we believe that the following additional requirements are appropriate:

- Full client-server system with fat server (SmartSkim engine) and thin client (SmartSkim interface) to exploit powerful server technology;
- Persistent storage of inverted files, to enable efficient relevance profiling of documents based on pre-existing inverted files (with automatic update when documents are modified);
- Shared access to inverted files in memory to reduce memory requirements

Persistent storage of inverted files for individual documents may be useful for other purposes, and especially given we index at the word position level. It could enable rapid phrase searching of individual documents. Given that documents are indexed individually, it would be relatively easy to build bespoke and customized search services over selected collections of document by combining the inverted files.

## 6. CONCLUSIONS

In this paper, we have introduced the concept of relevance profiling, and shown how language models can be used to generate relevance profiles. The SmartSkim tool, which implements relevance profiling, provides content-based browsing of documents, and effectively integrates concepts from passage retrieval and query-based document overviews such as TileBars. SmartSkim differs from TileBars by providing an overall indication of relevance for each text tile, and thus focuses on engaging the user with the text material itself rather than with the document overviewer. In this respect, SmartSkim is similar to the SCAN system, which provides an analogous profiling capability for audio documents. Importantly, we have shown how language modelling can provide an elegant and

simple solution for generating relevance profiles for text documents. The design of SmartSkim is discussed in detail, and we critique the design using (parts of) the Cognitive Dimensions Framework. Possible applications of SmartSkim are described in detail, and we consider how an operational version of the tool might be configured.

Clearly, experimental work is needed to evaluate the underpinning relevance profiling technique, and the useability and effectiveness of the SmartSkim tool from a user's perspective. Here, we will concentrate on the proposed end user experiments. Initially, we propose testing the tool for two different tasks, namely judging the relevance of documents based on relevance profiles, and using the tools to retrieve facts and answer questions. We will use the experimental protocol established for the TREC Interactive track, namely the interactive variant of the question-answering task. The experiment will be designed to study the macro-performance of the tools for those tasks, and to gain insights into the design of the interface by appropriate think-aloud experiments and by recording user sessions. Two interfaces will be test and compared, these being the SmartSkim interface, and a de-tuned version of that interface. The de-tuned version will not include the relevance profile meter, and instead navigation buttons will be provided for moving between highlighted query words. This will provide a slightly more advanced version of the typical "Find" function found in web browsers. This will be a severe test of SmartSkim, and will test the effectiveness of relevance profiling for identifying relevant passages within retrieved documents.

We believe that relevance profiling based on language modelling, could be extended and applied in a number of ways. Relevance profiling could be used as the first stage of an automatic question-answering or information extraction system. It would be interesting to integrate SmartSkim with a document reading appliance, such as XLibris , and explore the interaction between content-based document skimming of SmartSkim, and the annotation and querying functions of XLibris.

## 7. ACKNOWLEDGMENTS

We acknowledge the assistance of Gheorghe Muresan, who provided invaluable feedback on the paper, and who assisted in supervising the associated student projects.

## 8. REFERENCES

- [1] Hearst, M. A.: TileBars: visualization of term distribution information in full text information access. Proc. CHI'95, (1995), 56-66.
- [2] Whittaker, S., Hirschberg, J., Choi, J., Hindle, D., Pereira, F. and Singhal, A.: SCAN: Designing and evaluating user interfaces to support retrieval from speech archives. In Proceedings ACM SIGIR '99. ACM Press (1999) 26-33.
- [3] Kaszkiel, M. and Zobel, J.: Passage Retrieval Revisited. In: Proceedings of the Twentieth International ACM-SIGIR Conference on Research and Development in Information Retrieval, Philadelphia, July 1997. ACM Press (1997) 178-185.
- [4] Kaszkiel, M.: Indexing and Retrieval of Passages in Full-Text Databases, PhD thesis. RMIT Computer Science Technical Report (RT-17), May 2000 (2000).
- [5] Kaszkiel, M., Zobel, J. and Sacks-Davis, R.: Efficient Passage Ranking for Document Databases. ACM Transactions on Information Systems, Vol 17, No. 4 (1999) 406-439.
- [6] Landauer, T., Egan, D., Remde, J., Lesk, M., Lochbaum, C., and Ketchum, D.: Enhancing the usability of text through computer delivery and formative evaluation: The SuperBook project. In: McKnight, C., Dillon, A., and Richardson, J. (eds): Hypertext: A Psychological Perspective. Ellis Horwood (1993) 71-136.
- [7] Marchionini, G.: Information Seeking in Electronic Environments. Cambridge University Press, Cambridge (1995).
- [8] Byrd, D.: A Scrollbar-based Visualization for Document Navigation. In Proceedings of ACM Digital Libraries 99. ACM Press (1999).
- [9] de Kretser, O. and Moffat, A.: Effective Document Presentation with a Locality-Based Similarity Heuristic. In: Proceedings of the Twenty Second International ACM-SIGIR Conference on Research and Development in Information Retrieval, Berkeley, August 1999. ACM Press (1999) 113-120.
- [10] Tombros, A. and Sanderson, M.: Advantages of Query Biased Summaries in Information Retrieval. In: Proceedings of 1998 ACM SIGIR Conference on Research and Development in Information Retrieval (1998) 2-10.
- [11] Ponte, J. and Croft, W. B.: A language modeling approach to information retrieval. In: Proceedings of the 1998 ACM SIGIR Conference on Research and Development in Information Retrieval (1998) 275-281.
- [12] Song, F. and Croft, W.B.: A general language model for information retrieval in Proceedings of the 1999 ACM SIGIR Conference on Research and Development in Information Retrieval (1999) 279-280.
- [13] Green, T.R.G.: Describing information artifacts with cognitive dimensions and structure maps. In: Diaper, D. and Hammond, N. V. (eds.): Proceedings of the HCI'91 Conference on People and Computers VI. Cambridge University Press, Cambridge (1991).
- [14] Hendry, D. G. and Green, T.R.G.: Creating, comprehending and explaining spreadsheets: a cognitive interpretation of what discretionary users think of the spreadsheet model. Intl. J. of Human-Computer Studies, 40 (1994) 1033-1065.
- [15] Nielson, J.: Hypertext '87 Trip Report. ACM SIGCHI Bulletin 10, (1998) 27-35.
- [16] Schilit, B. N., Golovchinsky, G. and Price, M. N.: Beyond paper: Supporting Active Reading with free-form digital ink annotations. In: Proceedings of CHI98, ACM Press (1998) 149-156.
- [17] O'Day, V. L. and Jeffries, R. Orienteering in an Information Landscape: How Information Seekers get from here to there. In: Proceedings of INTERCHI '93, ACM Press (1993) 438-445.