Technical Report

UCAM-CL-TR-655
ISSN 1476-2986

Number 655

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Wearing proper combinations

Karen Spärck Jones

November 2005

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

ISSN 1476-2986

Wearing proper combinations *

Karen Spärck Jones

Abstract

This paper discusses the proper treatment of multiple indexing fields, representations, or streams, in document retrieval. Previous experiments by Robertson and his colleagues have shown that, with a widely used type of term weighting and fields that share keys, document scores should be computed using term frequencies over fields rather than by combining field scores. Here I examine a wide range of document and query indexing situations, and consider their implications for this approach to document scoring.

1 Introduction: Combining index types

There are many different sources of information about document content that may be helpful for retrieval purposes. They are conventionally represented by the different fields on a catalogue card (author, title, classmark, etc), extend to abstracts and citations, and have more recently been supplemented by full texts, URLs, and so forth; they also include other items like publisher, journal name, or publication date. As has long been recognised, all of these *index key types* say something more or less directly, more or less reliably, and more or less conveniently, about what a document is about. Their relative utility has been reflected in retrieval systems by such older and cruder mechanisms as specifying that search terms must occur in titles, rather than anywhere or, as typically in Web engines, by some sort of key type weighting scheme of which the user may be entirely ignorant.

In some cases there can be no key overlap between fields, for example between an image and its text description. However in others there are perfectly natural overlaps, for instance where a word is used, with the same meaning, in a title and an abstract. In this situation the status of a field weighting scheme may be more problematic because of cross-field dependency. In general, therefore, we have to consider the consequences of the way query-document matches on multiple fields are handled to arrive at a soundly-based final single document score.

The topic of this discussion is closely connected with, or has been approached through, several others, notably combination of evidence in retrieval (see e.g. Croft 2000), the use of query structure especially Boolean structure (e.g. Cormack et al. 2005) retrieval for structured documents (e.g. Fuhr et al. 2005), and also dealing with multimedia, especially text and image, material. The general issue is how to treat multiple types of information in documents as a joint signal of potential relevance, and in particular how to do this consistently within a modern statistically-based relevance ranking scheme. Croft provides an all-embracing view, achieved by significant, albeit effective, abstraction from detail. There are nevertheless important issues of detail connected with the way different information types or *fields* are related to one another and hence should be handled jointly. This paper approaches combination of evidence from this

*I am grateful to Stephen Robertson for stimulating this paper and for discussions.

point of view, taking Robertson et al. (2004) as its starting point.

'Fields'

The right way to treat information types is independent of whether the basic situation is described in terms of multiple indexing *fields* (as in Robertson et al. 2004, hereafter referred to as CIKM04), or multiple indexing *representations* (as in Turtle and Croft 1990 and Metzler and Croft 2004), or multiple indexing *streams* (as in Strzalkowski et al. 1999). Thus it is not immediately obvious that there is any fundamental difference between the case where the same key appears, just as it is, in different fields like title and abstract, or appears in closely related forms, for instance as a word in the title and an item in a Keyword list, or as a single word in one part of a document indexing representation or stream and as a component of a phrase in another (as in e.g. Mitra et al. 1997 and Strzalkowski et al. 1999). In the first case we appear to be refering to the document's physical fields, in the third to its logical ones but, as the second case suggests, this is not a fundamental distinction. Again, with conventional documents, title and text come from the author as part of the initial document, while standard subject headings can come later, as an index addition, but these are equally treated as generating index types when it comes to searching. So thinking in terms of physical fields is a mistake, and a more abstract view, of different representations or streams, is more appropriate in relation to the underlying (?Platonic) entity we take a document's content to be.

Taking types as decoupled from any particular source document layout is useful because it allows us to see different index types as different *annotations* on the same document content, which may in some cases be physically localised, as when a single word and a phrase are co-located in text, but need not be so, as when an assigned Keyword is applicable to a document as a whole. Indeed abstracting in this way serves also to make the point that while we may think of a title or abstract as part of a document but a subject heading as an addition to it, a title is really just one *manifestation* of the document's content and the heading is another, with the text body as yet another. Thus for each manifestation there is an associated index key type. For instance, when we treat the full-text field as a manifestation of the document's content for retrieval, what this actually means is using such an index key type for it as a set of single terms. With the older style approach to cataloguing it was evident that the fields in the catalogue record functioned as manifestations. With the newer situation of working directly with source documents it is more natural to think in terms of physical areas of the source. But the real situation is the same, and the important point is to consider what the implications of exploiting multiple fields, i.e. key types, at once are.

'Combination'

The same general issue, namely of how multiple inputs from different fields together for a single query-document score, arises whether the process is refered to as *combination*, or *merging* or *fusion*. A common approach to is to treat it as combining the separate matching *scores* for each stream to give one overall score. Then if one field regarded as more important for matching, this is marked by upgrading the score for this field before combination. So, for example, if we have three fields, title, abstract and subject headings, and regard a match on an abstract as more, say twice as, important than one on either of the other two fields, we score for term matches on each of the three fields separately, double the score for the abstract, and combine the score for the three fields for the final score determining the document's retrieval status value.

The details may of course be quite sophisticated, but there is an underlying assumption that it is legitimate to combine the matching scores for the individual fields, and that where we

want to vary the importance attached to different fields, the main issue is to find an appropriate weight for each field.

But this score combination strategy has three major defects. First, it ignores key type dependencies, and also, where key types are distinct, what follows from their particular characteristics. Thus with a rich range of document representations like that envisaged in Turtle and Croft (1990), we may have some representations, for example, title, text body, figure captions, that may share terms, and others that are disjoint, for example text and images. The presumption with dependencies is that even where the same word appears in more than one field, and with the same intrinsic meaning, there is an additional contextualisation, or valorisation, that is significant for retrieval and is, moreover, appropriately captured by 'double' scoring. At the same time, the scores for several such fields may be combined with that for the fundamentally different image field.

Taking fields as supplying distinguishing contexts when the same term appears in different fields may appear plausible, but it is not satisfactory because there is a real, if underlying, dependency between the fields: the same term appears in them, and for the good reason that this has something to do with what the document is about. Thus behaving as if term `T1` appearing in field `F2` is completely independent of the fact that it has already appeared in field `F1` is a fundamental error. This is regardless of the way scores are computed, and applies even in the simplest case where we record only term presence in a field.

Systems typically, however, go far beyond the simple term presence/absence approach, and compute scores involving term weighting. The second problem about the usual score combination strategy is that it ignores the consequences of the way the individual field scores are computed for the score combination. Most systems use a term weighting formula of a $tf * idf$ type. It is therefore important to consider what effects this may have when keys may occur in more than one field. But it is also sensible, in general, to pay attention to the impact of any field score's composition and rationale on subsequent score combinations. In particular we may ask whether combination may be more rationally based only on the raw *data* from which a field score is computed rather than on the derived field score: for example, whether we should use tf directly in some way.

The third problem comes when fields themselves are weighted so, for example, a match in the document title field is viewed as more significant, and hence should count for more, than one in the text body field. If the component score / overall score relationship is problematic even where all fields are accorded the same importance, it is likely to become more problematic when we consider variable field weighting; and weighting fields themselves differently is one of the main rationales for maintaining field distinctions.

'Document structure'

Talking about document (or record) fields assumes a notion of *document structure* (or *structured document*), which places the issues to be considered here in new lights. It draws attention, in particular, the roles of different components both in relation to a document whole and to one another, and to the possibility of complex document structures that may combine hierarchical relations (e.g. section, subsection) and non-hierarchical ones (e.g. quotation, regular text).

Making the connection between fields and document structure explicit, we may start by saying that 'structured document' refers to a document, or document surrogate like a catalogue record, that has multiple component *fields*. For example title, abstract, text body; or author, title, publisher. These fields are defined over the physical form of the document or surrogate. They may be explicitly given with the initial source e.g. /para, or mechanically determined for it e.g. 'sentence' in transcribed speech, perhaps dynamically at retrieval run time e.g. text

window. Fields may also be viewed rather as applied to, or attached to, an underlying physical source, rather than as an initial part of it, e.g. semantic category label on a text.

Document structures may be simple field sequences e.g. sentence 1, sentence 2, or nested e.g. article, headerdata, title; there may also be multiple structures associated with the same document.

Document structure in general can thus refer to *any* structure as defined by any set of annotating labels, with any label motivation and semantics e.g. discourse structure as defined by Rhetorical Structure Theory. The discussion here focuses on the types of structure and field which have traditionally been thought of interest for document and text retrieval e.g. title, abstract, citation, references, along with more recent fields like paragraph, anchor text. However the arrival of machine-readable text with e.g. HTML or XML tags has introduced many new tag types, often with much finer grain, of potential retrieval interest e.g. /ital or /emph.

Fields refer to format, which may or may not have content implications. The content connection may be close or remote. For example with bibliographic catalogue records, title and keywords fields have an obvious and direct content connection, author fields an indirect one, date fields (even when viewed as boundary conditions) a yet more indirect one. Among more modern types of field or tag, some again have a closer connection with content than others, e.g. (in general) anchor text more than URL more than /caps.

As some of the examples above suggest, there is great variation in whether the material within a field is likely to coincide with term matching for retrieval. Thus (in the past) a whole abstract text has been unlikely to be coterminous with a search query. But an author surname, or subject category label, may be, as may a (previously tagged or dynamically identified) text phrase. In general therefore, with respect to retrieval having an interest in document content, an expanded notion of document field emphasises the shift that automated, natural-language retrieval prompted, namely from thinking of document indexes, descriptors, or representations as something independent from the actual document itself, to taking the document, or a component of a document like a word, as its own description.

This ambiguity is well shown in Metzler and Croft's (2004) discussion, for the inference network retrieval model embodied in the InQuery system, of the status of representation nodes. InQuery allows for document nodes (DNs), text representation nodes (TRNs), and representation nodes (RNs). DNs are abstract entities - which we may think of as having a unique identifier; which are embodied in TRNs, where TRNs are most obviously what we can call these *complete* texts referring to the whole document (since "full text" is ambiguous between this and the text body apart from front matter etc). RNs are then individual representative elements referring to documents, for instance specific index terms. In the InQuery general model all of DNs, TRNs and RNs can be of different types. Thus a DN, TRN and RN could actually be image-based not language-based. Metzler and Croft comment on the fact that in the earlier view of InQuery, TRNs were taken as conflated with DNs, but that when language modelling is added to InQuery, it is more appropriate to separate DNs and TRNs and to conflate TRNs with RNs.

The recent growth of interest in the status and use of fields in retrieval has several sources. The most obvious is the stimulus provided by formatted document data, as embodied in Web pages and exploited in Web retrieval evaluations (Hawking and Craswell 2005) or exemplified by the spread of XML and associated XML-based retrieval programmes like INEX (Fuhr et al. 2005). This information may be used in various ways, on the one hand to allow searching on particular types of key or on the other to modify the value of plain word matching. The former is associated with a more general, growing interest in providing more focused retrieval returns than whole document texts, e.g. by returning paragraph-length passages. The latter is related to the continuing interest within mainstream text retrieval in using fine-grained terms, notably through various ways of defining or approximating syntactically and semantically well-grounded

multi-word units. There is yet another source, in the belief that more discriminating query-document scores can be obtained by factoring in different types of match context, i.e. field, types, e.g. by assigning more value to a within-title word match than a text body match, while recognising both, or by using both within-phrase and single-word term matches for queries.

There are clearly complex similarities as well as differences between these three. Much depends on the nature of the document. Thus in some cases the material in different fields in necessarily completely separate e.g. one has an image, another some keywords. In other cases different fields may overlap e.g. they use English words in the same natural way. But clearly what matters is why field information is being used. Thus we can distinguish

(1) specifying particular fields as match localities, in searching that will return whole documents. In the simple case there is one such field, or alternatives where only the one that matches best determines a document's score. Example: confining matching to the author field, or to the title field.

(2) enabling finer output ranking for whole documents by exploiting, and in particular combining, component information for term matches but without requiring component naming in the query. Example: scoring highly for a term match in a title and also, but less so, for a text body match.

(3) allowing multiple document components to be retrieved as separate items in search output, i.e. retrieval is at the sub rather than whole document level. The components are typically of the same type, but need not be. Example: returning paragraphs.

There is, however, no necessary distinction between these three. It depends entirely on what the form and relationship of document fields is, i.e. on the material in them and whether there is field overlap, for example through nesting.

Approach

In the rest of this paper we examine the issues raised above in more detail, for a range of indexing and matching situations. We will usually refer to fields and their index key types, but the points apply equally to representations and streams. The paper takes CIKM04 as its starting point, and considers what happens when we have more complex indexing and retrieval situations than the long-familiar one represented by 1) flat, single-field documents, 2) natural-language terms, 3) simple single-word terms, 4) statistically-based term weighting of the $tf * idf$ type (and also, subsidiarily, 5), retrieval from a single file). Such models as that presented in Spärck Jones et al. (2000) for this case reflect a single type of document representation and form of evidence for (probable) document relevance to a query. The question is therefore how to proceed from this, in a principled way, to the treatment of multiple forms of evidence that may be explicitly or implicitly related to one another in complicated ways, but without undue abstraction.

This development is not limited to the document point of view I have so far taken. It is also important to address the query point of view, once we abstract away from the idea of physical fields and consider multiple representations of underlying objects. Thus we may envisage multiple representations of the user's information need, even if a document has only a single representation, as for example in Belkin et al. (1995) and Cormack et al. (2005). We may also have multiple representations of both document and query - see Turtle and Croft (1990) and Metzler and Croft (2004). The issues about multiple key type combination thus bear as much on queries as on documents, and I will examine the query implications after analysing the document ones. In particular, while in practice $tf * idf$-type document-based weights are only computed for query terms and thus might be thought of as query term weights, it is also possible to take within-query term frequency or other query-based term weighting into account,

including user-determined importance weights assigned to terms in addition to whatever other contributions are made to their matching value, as in Liu et al. (2004).

The analysis which follows is deliberately informal and 'bottom up'. Some approaches to these issues are within formal frameworks, e.g. Turtle and Croft's (1990) treatment of representations within the framework of probabilistic inference nets and Croft's more recent development of this approach as evidence combination (Croft 2000). This generalisation and abstraction is valuable, but the need remains to consider the semantic character of fields and their relationships. For the same reason, though it may be appropriate to apply machine learning to derive field weights, as in Zaragoza et al. (2004), it is still helpful to consider the underlying rationale for using the fields and the semantics that motivate the emergent field values.

Questions of how to deal with mixed data types have also become more pressing with the arrival of multimedia files, and structured document retrieval has become an active research area not only because structured documents exist on the Web but because of the spread of structure formats like XML (e.g. Fuhr and Grossjohann 2004). The issue again here, however, is with how different field values are handled as relevance indicators, both when fields are disjoint and when they are nested, rather than with the mechanics of document processing. However new forms of text document that are appearing, like email with embedded text quotes from previous messages, which also raise field treatment issues, as illustrated in Frommholz (2005), where quotes are downweighted to null status.

Thinking generally about fields, and moving away from physically separated fields towards logical fields or representation *types*, also allows for the possibility that there may be multiple instances of a type and, further, that these type instances may not be 'detachable'. Thus while we may choose to see a title as a field for a document as a whole (even if it has multiple term matching units within it), it makes less sense to treat internal structural objects like sections as detachable. They do not form a single section field, but multiple distinct section fields; and such fields may have specific relationships with other fields, as in hierarchically structured documents. Searching still applies within these separate field instances just as for the case where there is a single field like a title.

The first part of the discussion which follows deals with issues that arise regardless of whether there can be only one or several fields of the same type, implicitly assuming that multiple fields of the same type will all be treated in the same way. However it is perfectly possible to give different instance fields different status so, for example, document opening sentences are treated as more significant than any other sentences. This is implicitly sub-typing them, It is also possible, as with Lalmas and Rölleke (2004), to give different instance fields different weights within the same document, e.g. to say subsection 1.1 is more significant for section 1 than subsection 1.2 is, whereas the reverse applies for 2.1 and 2.2 within 2. But these refinements do not necessarily make a logical difference as opposed, possibly, to an implementational one.

The initial focus is also on *document* fields. But as modern approaches to retrieval, bearing particularly on term weighting, have shifted document indexing from file time to search time, so the form of queries in the context of field matching and weighting has also to be examined. The analysis in the next section is deliberately presented through very simple examples, to lay out the issues clearly.

2 Key type cases

Case 1: Multiple fields, shared keys

We first consider the case where we have more than one field, but only one instance of any field type, and the same key may occur in more than one (indeed all) fields. For example we have title,

abstract, keyword, and text fields and the same word can appear in all. More modern examples, illustrating the logical rather than physically distinct fields, include the collected anchor text for hyperlinks when viewed as significant for the source document. For convenience we assume simple single words as terms, but the keys can be any natural language units that are treated as undecomposible for any representation or matching operation.

In this situation it is natural, as just mentioned, that the matching score for each field should be of the same sort and, more particularly, be based on a term weighting function of some familiar $tf * idf$ kind with appropriate document length (dl) normalisation, as with the BM25 function. Then for the same term a occurring in all four fields, we may have four different tf values. However if we assume that all field types occur in all documents the file size N is the same for all terms and idf remains document based; and if we regard the indexing as all relative to the document, we have a single dl.

As CIKM04 shows, this approach is unsatisfactory for any formula which does not use tf raw, but treats it in non-linear fashion, so that as tf increases there is progressively less value in the information a new term occurrence supplies. This implies that where the same term occurs in several fields, it has a set of tf-dependent weights that collectively count for more than a tf weight computed over all the occurrences together with the field boundaries ignored. But it is not clear that the term deserves the degree of emphasis that follows from field separation, even before compounding its effect by using field weights. The CIKM04 proposal is therefore that the only output for a term per stream should be its tf, and that field weighting should be applied in the form of field repetition. Thus if the title is weighted 2, the material in the title field is duplicated. The importance of the title and of the terms in it is thus reflected in higher tfs for the terms concerned, and there is in fact no need to preserve the field boundary. The net effect for the whole document, when all fields have been appropriately fixed, is to obtain a final *version* of the document in which some terms occur more frequently than they did originally, and which is also somewhat longer than the original (logically speaking, that is: the source document is not actually modified, only the indexes into it). The matching score for the document as a whole is then obtained in a completely straightforward way by using the new tf values.

These new tf values have, of course, also to be normalised for document length dl. In CIKM04 this was treated in a quite straightforward way using the whole document length after extension to reflect increased field lengths, and the length-related parameters b and k were also adjusted accordingly. However since (as discussed further under Case 2 below) there may be wild variations in field length, and especially systematic variations, the right treatment for tf, i.e. what differences in tf signify, needs to be less crude. We have to allow for the possibility that it may be desirable to take into field differences in dl into account, even though the same key type applies to all. For example if we have a title field and a full text field, the former is likely to be very short and the latter much longer, so one might want to give not just an extra weight to a match on titles, using the replication mechanism described, but also factor in a corresponding modification of a term's title tf by its title dl. The BM25 formula is restructured to allow this in Zaragoza et al. (2004), hereafter referred to as TREC04: dl is tied to field type, and tf values per field (renamed $pseudo - tf$ values) are adjusted by field dl. In TREC04 initial rather than extended field lengths were used, without untoward effect because the ratio $dl/av(dl)$ remains the same. However using the new expanded field dls would be more transparent and also avoid potential improprieties in computing scores further down the line. Using individual field dl also requires field-specific settings for the BM25 parameter b, as discussed in TREC04, and implies change to the k parameter: in TREC04 this is geared to averaging over the document fields, taking into account new field lengths, as is reasonable when keys are all of the same type across fields, but it might in principle be more properly related to individual fields.

It could also, with very heterogeneous files, be appropriate to allow for different values for the

file size N in relation to different fields and computing *idf*s for terms in them. For example, with a file consisting of subfiles, one with a full text field and the other without but both with titles, we could have an N value for the text field smaller than the total N applicable to titles. In both CIKM04 and TREC04 it is assumed that all fields occur for all or near enough all documents, so *idf* is handled in a quite straightforward way. In general we might think of tying all term weight components to specific fields, i.e. work not only with field *dl*s but also with field-specific *df* (and hence *idf*), by attaching appropriate *tags* containing this additional information to key tfs. However making *idf* field specific presents problems for the BM25 approach when *idf*s are combined over fields, in the way that field-specific *dl*s do not. Field-specific *idf*s cannot be simply incorporated, replacing global *idf*, and trying to get round this by using e.g. $av(idf)$ for a term are clearly ad hoc.

Note that where the field tf values are all 1, the CIKM04 strategy always defaults to a single document; on the other hand where field *scores* are combined there is no corresponding consistent default, even where the field tfs are all 1.

[It might seem appropriate to replace the original document-based tf, $(i)df$ and dl by tff, $(i)ff$ and fl respectively, but since I shall continue to refer to the basic BM25 formula I will continue to use the original tf etc with modifiers were required, as in "field tf".]

Example

It is useful to illustrate the basic points of the CIKM04 argument in relation to tf. Suppose we have a query and two documents with 3 fields and terms occurring as follows, i.e. terms can occur in more than one field; and further we initially assume that we are in fact only interested in whether query terms are present or absent in the document (i.e. there are fields but we are not interested in exploiting the differences between them for scoring in any way).

```
Q :  a b            D1:  F1:  a
                         F2:  b
                         F3:  b

                    D2:  F1:  a
                         F2:  a b
                         F3:  a b a
```

The first point is about the correct treatment of matches on fields, which can be illustrated simply with term presence/absence: no weights. Then if we match `Q` as a whole, i.e. terms `a` and `b`, against fields, we will not get a match at all for `D1`, but match, in fact twice, for `D2`; however we only count the match once, to obtain the two terms present in the document, giving us a document score of `2`. But we have not specified that the query `a b` has to be treated as an indissoluble unit. Thus an alternative strategy is to match first `a` on all fields, and then `b`. In this case both `D1` and `D2` match, with the same score of `2`.

The second point is about the correct treatment of scores for fields, and specifically about the effect of tf. The argument here is that with any scoring function involving tf but not linear in tf, and BM25 is such a non-linear function, it is improper to compute scores for each field and then combine them. The proper strategy is to obtain tf for each field where a term occurs, and then, adding these, to compute the overall matching score. In the type of situation envisaged in CIKM04, where a term can occur in any field, all fields occur in all documents, and there are no systematic differences in field lengths, *df* and hence *idf* is over the whole document, and it is also appropriate in normalising tf by *dl* to use the whole document *dl*. Essentially what we are doing, in computing scores, is abandoning the field boundaries. Thus for the example query, `D1` has $tf = 1$ for `a` and $tf = 2$ for `b`, and `D2` has $tf = 4$ and $tf = 2$ for `a` and `b` respectively.

The third point is that if we now decide to give field `F2` a weight of `2`, say, this doubles the tfs of all the terms in that field. So the final tf for `b` in `D1` becomes `3`, not `2`, and the tfs for `a` and `b` in `D2` become `5` and `3`, not `4` and `2`. This is the only way to handle field weighting given the second point about the way to handle tf.

The net result of all of this is that we start with an initial document with explicit fields and proceed to a final document in which the fields are all merged, but the value of any different original field distributions for terms preserved. The contributing frequency information for terms is gathered but matching scores are not prematurely computed.

CIKM04 demonstrates, very clearly, that using field data, not field scores, in combination in the way described gives much better (indeed surprisingly better) retrieval performance than using field scores, where the fields used can freely share keys: in this cases title, text body, and anchor text drawn from citing documents. These field clearly share keys (though not necessarily the same stop list), but the keys have different distributions in the three. The CIKM04 strategy also has a very useful practical payoff, since the whole can be implemented extremely simply. The replacement of overall *dl* by field *dl* in TREC04 is the appropriate natural extension of CIKM04, reflecting the close relationship between term frequency and unit over which this frequency is computed.

Since this way of dealing with fields improves performance, it is presumably capturing something of what field differentiation as a way of giving term occurrences different values is meant to achieve. Thus while, for the term situation described, we may assume that term `a` in the title has the same primary meaning as in the abstract, the significance of its occurrence in the title, which contextually enhances its status as a content indicator for the document, is adequately captured by increasing its frequency. What we are not capturing, since the field boundaries are removed, or at least are only indirectly or partically capturing, is any significance in the term's occurrence in a capsule, highlighted part of the document as a discourse. But maybe this is not really necessary.

Case 2: Multiple fields, different term-like keys

Here we consider keys that have frequency properties like natural language terms but have their own individual fields. There are many examples both of conventional and newer kinds. For example, especially where publication conventions force it, kinds of key occurring in bibliographies like journal names, or journal volumes, or publishers. Other examples are numerical class marks or subject codes, from conventional document indexing. Where retrieval strategies treat cited URLs in a distinct way, rather than as if they were text words, these will have their own logical field which may be favourably weighted, as for example in a Web engine.

In principle these types of key may have tfs other than 1, for example the same journal may recur in a bibliography, and the same could be true for cited URLs, though others may necessarily only occur once per document.

As these key types cannot occur in multiple fields, there is no reason to avoid the field score problem for them. But once the strategy has been applied of not computing scores for some fields, they cannot be computed for others as this would create an entirely inconsistent base for computing final document scores. The independent reason for treating such fields in the same way as the original one is that this makes it possible to apply field weights in a completely straightforward way; indeed not doing this, and retaining the use of scores, when a field weight is greater than 1, is tacitly having multiple identical fields for a particular key type and then combining their scores with the original untoward effects.

Cases 2, more than Case 1, prompts thought about the treatment of *dl*: because the keys in the different fields are not of the same type, we should not just assume that a single *dl* applies across all fields. For example suppose we have a full text field and a field with the 'up to three' Keyword or subject headings that are often assigned to scientific papers. The limiting case is where, for example, indexing policy assigns only a single class mark to a document, or where a particular key type can naturally occur only once for a document, for instance a document's own URL. It is not clear that *dl* normalisation by the entire document length, say 1/3467, is appropriate since this drastically downweights such keys. But it is also not clear that normalisation by just the field length is appropriate either: this would give 1/1 for the URL example, thus overemphasising the key's contribution to a document's overall matching score. Field upweighting within the CIKM04 framework does not alter the general argument: for example upweighting the abstract field so a term occurring in it counts for more would still leave a Keyword match much more valuable; and nor does the fact that in a formula like BM25 raw *dl* is normalised by *avdl*. In fact, as we have seen, the same situation can arise under Case 1, prompting the adoption of field-specific *dl* and adjustment of b and k and stimulating the application of machine learning, as described in TREC04, to determine not only settings for these parameters but for the field weights themselves. There are still problems when individual fields can vary wildly in length, perhaps without document content motivation, as with spam stuffing of Web page fields, as discussed further later.

Where all the fields, i.e. key types, apply to all the documents in the file, we can reasonably use the same *df* base for computing *idf* for the various key types. But if the file was heterogenous, and key types could not apply to all documents (or, as mentioned under Case 1, if they were found differ systematically or widely in practice), it would be proper to allow the *df* bases for the different key types to vary, as suggested earlier and as done in e.g. Lalmas and Rölleke (2004) and Liu et al. (2004). We can thus envisage individual key tfs having attached pertinent field tags for *dl*s and for *df* counting in order to obtain $(i)df$s as input to score computation. This approach to $(i)df$ would not be a problem for any key type that was confined to a single field but it would, as noted earlier, present combination problems for key types that could occur in more than one field.

In effect, what is being done in Case 2, when taking field weighting into account, is moving toward *subdocuments* for each key type, after we have merged and possibly replicated fields within each subdocument. But after any field replication, there is no actual need to retain the subdocument boundaries. If the term types are seriously distinct, all the subdocuments can be merged because matching will automatically select for the correct type. However it is likely in practice to be helpful to retain type *flags* in the indexes to the documents to deal with the cases where the same symbol can occur in different fields but where it is to be interpreted as of different types. For instance the user may require that a person's name is treated as the document author's name, not a cited author's name. Thus we may suppose that a key in the the index for the document file has a number of attribute slots for flags.

This last example also draws attention to the fact that, whereas with Case 1 we assumed an undifferentiated query, for Case 2 we have to allow for key type differentiation in the query, and for type correlations between query and document; and indeed there is no reason in principle, as with XML queries (Liu et al. 2004), why the types should not be differentially weighted in the query, as considered later.

In general, there would appear to be considerable advantages in the decompositional approach to weight-based scoring advocated in CIKM04, since it makes what is happening with complex indexing and matching situations more transparent. Moreover, the compositional approach to obtaining a BM25-based score from several sources of information about a document, for Case 1 from field values and here from different key types as well as field values, carries over

formally to Case 2, though appropriate treatments for *dl* and *df* in the more complex Case 2 situation are also required.

Of course we cannot suppose there is *no* relationship between the keys of different types, for example in the occurrence of the same word in a document title and as a subject heading from some approved list of headings or person name in a text and in a bibliography, or in the appearance of a particular name in an author field and a particular word in a text. But this is no different from the dependence that already exists, at the meaning level, between different words in a single field. We do not know (precisely) what this dependence is, and we may deem it generally weaker than the dependence that holds for the same word being used with the same meaning (though not necesarily contextual implication) under Case 1. The Case 2 exclusion of relationship thus reflects the general presumption underlying BM25, and many other approaches to indexing and matching, that we can behave as if there is no term dependency in practice even if we acknowledge it is there in principle, a presumption that is justified by experimental findings.

Case 3: Multiple fields, non-term-like keys

The illustrative case here is that of museum catalogues or image databases, which typically have several word-based fields and also one with the actual image (or, perhaps, several with different images). The same situation arises with papers that have text figures or, we may imagine, files which have incorporated recorded music or (untranscribed) speech elements, as well as with video which may include a text caption stream. There may of course be several such fields of different types e.g. photographs, graphs.

The problem here is that the image field, for example, may have no keys in the ordinary sense. Strictly, just as with text, we are dealing not with the image or figure as presented for user viewing, but with some coded form of the image that is taken as input to any image retrieval procedure, and which may range from a pixel array to some derived surrogate like, for example, a colour histogram. In such cases, where image retrieval might be done by submitting an example image, the image processing for comparative purposes may be a highly complex process applied to image representations where there are no elements that can be viewed as terms, certainly not ones interpretable by users.

Where the surrogate is some special-purpose pictorial representation language - for example one consisting of basic shape primitives that a user would also exploit for queries, we have a version of the previous Case 2. But we have to allow for the fact that matching operations on image fields do deliver scores that cannot be decomposed in the style advocated for words in CIKM04, i.e. we have a more aggressive form of the subdocument situation than with Case 2. It would appear that the only way of obtaining a final single document score has then to be by computing a final score for all the non-image fields and combining this, essentially ad hoc after experiment, with the image field score. This would also be the case if, for example with searching using an example image, the score for retrieved images was thresholded for match/no match. This is still a score to be combined with the other non-image ones.

As a corollary, also, while the idea of upweighting a match on an image field makes perfectly good intuitive sense, it could not in general be handled by the replication strategy applicable to Cases 1 and 2.

Case 4: Overlapping keys, multiple fields?

We now come to the problem case.

This is the situation where we have keys that do not occur in identical form in different fields, but have some obvious, indeed *necessary*, relationship. There are clear natural language

examples, for instance where we have simple single terms, and also multi-word phrasal terms, and where (as in Mitra et al. 1997) the field indexing for single terms can include occurrences that are also occurrences of the term as a phrase component. Strzalkowski et al. (1999) offers another form of this key overlap by using full word forms and stems (as well as phrases) as separate streams. Experiments hitherto with phrases have suggested that it is essential to allow individual component word matches where full phrase matching fails, since this is capturing something of the concept sought. However allowing all terms to occur as single terms as well as, where appropriate, to figure in phrases, implies that where phrase matching succeeds the component single terms are also independently matched. This might just be a byproduct of some reluctance, for practical reasons, to record genuinely separate occurrences of single terms as distinct from phrasal occurrences, even though the two are related. But it is primarily a matter of policy, reflecting the difficulties of ensuring compound term matching and justified because the variant 'duplicates' can be seen as a useful reinforcement of the concept involved. The implications for the 'field' view of documents are in any case the same, and thus should preferably be explicitly examined rather than lost in low-level implementation detail.

The way that single terms and phrases are handled as separate vector areas in the SMART system (Mitra et al. 1997), and as separate indexing streams in Strzalkowski et al. (1999), assimilates the indexing and matching situation to Case 2. But the situation is not like those considered under Case 2, since the indexing languages for the different fields are not distinct: there is *necessary* dependence between the 'variant' terms occurring in different fields, i.e. a much stronger form of dependence than that illustrated by whatever relationship there might be, under Case 2, between an author, or a subject class mark, in their fields, and the words used in an associated full text field.

Thus if we allow matching on both single term and phrase fields we are automatically upweighting the common concept, and indeed may have further upweighting through the effects of phrase rarity on the *idf* component of weighting. This has been observed in past experiments to lead sometimes to document 'overpromotion'; and one response has been to vary the field values for single terms and phrases respectively. But in general, past experiments have not considered the ramified effects of combining different fields of this logical kind with fields of the other physical kind, like titles and abstracts. The presumption, with natural language indexing, would seem to be that if there is a title single term field, there also has to be a title phrase field, and so on. It is easy to see that the underlying concept repetition on the one hand, when combined with field weighting on the other, could have rather large effects on final document scores.

There is no mechanical problem about all this: the issue is what justification, both in general principle and in relation to the assumptions underlying BM25-type retrieval strategies, can be given for treating overlapping key types in the distinct Case 2 style. (It follows, of course, that there have to be query fields for appropriate term type attachments and, as with Case 2 in general, we might want to allow variable *dl* and *df*.)

The alternative approach is to ignore the necessary relationship between phrases and their component single terms, or between full words and their component stems, and regard it as more like the contingent relationship between synonyms or hypernyms/hyponyms: we are well used to having terms with these kind of relationships all thrown together as of the same single key type, for instance as single terms. In this case we assimilate the situation to Case 1, and handle fields and their weighting in the completely straightforward way described for Case 1, with tf etc similarly managed. We will still get multiple matches on the same shared concept - if a phrase in field `F1` matches then its component terms will, and this may count for more if `F1` has an extra weight, for example as a title. However what this Case 1 approach does not allow is any way of differentially weighting single terms and phrases as distinct key types associated

with logically different fields.

Again there is no mechanical problem here: the difficulty is at the principle and modelling level and their underpinning for the actual term weighting formula the system uses. Playing for safety by not allowing matches on term components as well as their full forms avoids the difficulties just described but conflicts with what seems to be needed for effective retrieval.

This ambiguity about phrases is reflected in Croft et al. (1991): different ways of defining phrases within the inference net model precisely reflect different views of the relation between phrases and their components, for example whether single term node information propagates through a phrasal node for a phrasal match, or whether single term nodes and phrase node all contribute in parallel to matching. In Metzler and Croft (2004) such related natural language expressions are all treated as alternative representations of content that can be freely combined. However the discussion in both cases does not cover the interaction between these 'logical' fields and physical ones like titles and abstracts.

The discussion has so far tacitly assumed that multi-word terms come from an independently established phrase set (and thus might be assigned at document file time). There is an additional complexity when phrases are defined dynamically through the query at search time through proximity and perhaps allowing variable interword distances (e.g. Keen 1992). One might imagine logical fields of, say, proximity distance 3, but these will not be physical fields, and so phrase upweighting has to be consistently applied to the tf component in computing term weighting for such phrasal terms. This is quite apart from the point about component term 'inclusion', which still applies.

The problem considered here for single words and phrases may also appear in other, perhaps complicated, guises. For example in the context of spoken document retrieval, Jones et al. (1996) use two ways of deriving words from the original speech and exploit both for indexing. The reason is that dictionary-based methods of speech transcription may not capture e.g. proper names, which can be matched with phone-level representations. However individual words may be found both ways, which may be deemed a form of Case 4 rather than a proper Case 1, though it may be treated as a Case 1; however if such duplicates are weeded we have Case 2.

3 The higher combination

The primary argument for the field management strategy presented in CIKM04 and its development in TREC04 was that defined by Case 1. As we have seen, it can apply to others, like Case 2, by natural extension. Some situations, like that just considered as Case 4, are more problematic. However there is still an argument for working with the decompositional approach to BM25(-type) weighting for fields - where fields are defined by the abstract notion of index type and thus can subsume both what are thought of as different key types like title words or assigned subject headings, and what are thought of as different indexing areas like title words and abstract words. This subsumptive mixing is well illustrated by conventional bibliographic systems but clearly also applies in more modern contexts like Web pages.

The discussion of the different cases in the previous section was primarily focused on the treatment of individual fields or particular subsets of fields. The effects for documents as wholes were thus only incidentally indicated, so this section illustrates the effects for a whole document.

Applying the CIKM04 approach with overall dl, suppose we have documents with 5 fields altogether, with languages and field weights as follows:

```
Field 1        title in English          Field weight 2
Field 2        title in French           Field weight 2
Field 3        abstract in English       Field weight 1
Field 4        publication date          Field weight 1
Field 5        own URL                   Field weight 3
```

To simplify we assume that there cannot be dates in titles or abstracts (this is equivalent to deeming that 'publication dates' constitute a distinct language), and that English and French do not share symbols (so there is no need to flag which for ambiguous symbols). Terms may clearly be shared between Fields 1 and 3, however. We also assume that URLs are undecomposible, but top-level components only (so they may apply to more than one document).

We first apply field weighting. Thus replicating fields with a weight greater than 1 gives us:

```
Field 1a       title in English
Field 1b       title in English
Field 2        title in French
Field 3        abstract in English
Field 4        publication date
Field 5a       own URL
Field 5b       own URL
Field 5c       own URL
```

We now apply the field merging strategy for fields with shared terms, so our document consists of:

```
Subdocument A     [Fields 1a, 1b, 3, all run together]
Subdocument B     [Field 2]
Subdocument C     [Field 4]
Subdocument D     [Fields 5a, 5b, 5c, all run together]
```

This represents an occurrence of the Case 1 situation for the original Fields 1 and 3, with Case 2 for each of Fields 2, 4 and 5. As part of the merging process we compute tf for all the terms, which we make available for subsequent scoring along with tags indicating whatever we decide should be their associated dl- (and perhaps df-) relevant data for the subdocuments in which they occur.

The four subdocuments are *virtual* subdocuments: they are distinguished by the frequency and associated weighting data for their respective term sets. We can now, in retrieval, apply BM25 to obtain the document's matching score. This can be done, indifferently, either by obtaining BM25 values for each subdocument and summing these, or by computing one global BM25 over the whole matching term set, since the distinguishing information for each subdocument is already reflected in the term data.

Clearly, if we have Case 3 as represented by an additional original Field 6, say containing some recorded music, then we will have to find some way of combining the score for a match on this with the BM25 we have obtained from whatever matches hold for Fields 1-5.

Now consider what we do if we have, say, phrases as well as single terms, i.e. we have a Case 4 situation. If we assimilate this to Case 1 we simply include phrases as well as single terms in whatever fields are pertinent, say the English and French language ones respectively and proceed as before. On the other hand, if we adopt the strategy of treating indexing with single terms and with phrases as using different languages (in spite of their dependency), we assimilate to Case 2. Thus the CIKM04 strategy would give us:

```
Field 1.1      title in English, single terms
Field 1.2      title in English, phrases
Field 2.1      title in French, single terms
Field 2.2      title in French, phrases
Field 3.1      abstract in English, single terms
Field 3.2      abstract in English, phrases
...
```

with the consequence that, for title field weights of 2, we will now have:

```
Subdocument P   [Fields 1.1a, 1.1b, 3.1, run together]
Subdocument Q   [Fields 1.2a, 1.2b, 3.2, run together]
Subdocument R   [Fields 2.1a, 2.1b, run together]
Subdocument S   [Fields 2.2a, 2.2b, run together]
...
```

An alternative more radical strategy, developing the CIKM04 approach, is to say that (when there is no Case 3 field) there really is no need to keep the different subdocuments apart, even if they involve different languages. They can all be merged because they are all *manifestations* of the same underlying document content - even URLs have some relationship to what documents are about. The fact that we may have wide differences in tf levels is neither here nor there. Thus once we have replicated fields to reflect field weights, we simply make a monster bag of words and compute $tf * idf$-type weights with field dl factored in, e.g. using BM25, for each key. We are after all assuming, throughout our approach to retrieval, that terms are independent even if they are not. Thus we just merge any fields regardless of whether they are of types 1,2 or 3.

The foregoing looks at the situation only from what may be called the 'what a document is' point of view, and also logically rather than implementationally. In retrieval matching only specific keys are of interest. This abstract view simply requires that, as mentioned earlier, the content of the tag associated with each key covers the specific information involved (dl, and the BM25 parameter settings for b and k). But if there is any likelihood of key symbol ambiguity across fields, the practical implementation must be more particularised.

4 Complications

Fancy field weights

The discussion so far, talking about field 'replication', has tacitly assumed that field weights have integer values. But of course this is not necessary, since there is no literal replication of parts of documents or their surrogates. Field values can seek to make more subtle adjustments to the relative importance assigned to different fields. Thus, as described in TREC04, field values can be learnt rather than simply assigned.

Term certainty

We have also so far assumed that whether a term is present or not is clear. But we must also allow for terms that are not simply present in a clearcut way, but have numerical values for their individual occurrences. For example we might have transcribed speech, or an OCRed document, which could have probabilities associated with each term occurrence. But factoring such primary

information about key occurrences into the basic data for terms is a general problem for weight computation independent of the field issue (see Jones, 2000), and quite apart from any practical complications (for example getting phrase probabilities as well as single term ones). As long as such information is tied to key occurrences, and hence to tf data, it is wholly compatible with the way of handling different field values advocated in CIKM04. Thus if we have term t with some complex modified tf, for example obtained by multiplying the individual occurrence probabilities, we can repeat this tf value when we replicate the field.

Hierarchical document structure

We have so far considered only flat document structure, though the sequence of fields in a document can be formally treated as a set of leaves all immediately dependent on a single 'docid' node. But hierarchical document structure is often much more substantial: for example chapter, section, subsection and, with modern document formatting languages like XML, hierarchical structure can be explicitly represented and made available for searching. There is, moreover, no reason why such structure should not be combined with statistical models of term weighting, as in Fuhr and Grossjohann (2004).

But structure of this nested kind is tricky. For the moment we ignore node *labels* i.e. type indicators or headings (e.g. 'section' or 'Background') as information entities in their own right, in order to focus on the material belonging to a node. Suppose then that we have a document with a node level representing sections and a lower node level below this representing subsections. From one point of view we can view a section - as flagged by 'section' - as a field, and take this as implying that its field content is all of the dependent text from all of the subsections. Each subsection as a field consists in turn of all of the text in that subsection. It then follows that there is a necessary overlap between material in superordinate and in subordinate fields. The alternative view is that 'section' is not an independent substantive field: it is no more than the material from its subsections when taken together from a second point of view, and is not a second independent entity in its own right. If it is only a second view of the same basic material, there can be no overlap with terms automatically in a higher field because they appear in a lower one.

The general retrieval argument for working with hierarchical structure is that a document which matches a query within a more confined *scope* is likely to be more relevant than one which matches only within a larger scope: thus if a query has an `a b c` match within a subsection this is better than a match with `a`, `b`, `c` on different subsections even though this is still within the same section.

In the examples we considered earlier there is no field subordination (the top-level 'document' is a formal superordinate). But what happens if we have field inclusion, like section/subsection or text/paragraph or whole-document/text-body? We have to consider what happens both in the system's internal operations on which we have concentrated so far and also when we have explicit field preferences: these may be applied automatically to produce the system's output or may be specified by the user. Allowing matches on title or on "anywhere" - title or abstract or remaining text -, is not a problem if these are two alternative specifications giving independent item (document) outputs, because there is no inclusion relation between the fields in the "anywhere" case. However if we allow matches on section or subsection, where section includes one or more subsections, we have problems not because of allowing alternative separate outputs but of how to compute the matching score for section given that it is made up of subsections. More generally, we have this problem if we simply specify any one field in a hierarchy that has one or more layers of subfields below it, and the same applies if we allow

'vague' field specifications, for example ones that permit matches not only on the specified field but one, say, up to two levels above it, designed to offer the user multiple outputs reflecting tighter or looser matches (see e.g. Lalmas and Rölleke 2005). How should data for lower fields contribute to higher ones?

The kind of hierarchical document field situation we are considering here has the same necessary overlap property as the phrases and their component terms case we discussed earlier, but the inference by analogy, that we should have both section material and subsection materials, and either throw them all together in Case 1 style, however repetitive, or alternatively rigorously separate them in Case 3 style, does not really carry through. The necessary overlap is much greater since there can be nothing in the section field which is not in some subsection field. Thus the independence assumptions that underlie BM25-style matching are much more comprehensively violated.

The proper treatment for hierarchical document structure, if we want to apply the CIKM04 approach in a reasonably kosher manner, is to work only with tree leaves. Thus for the example two-level structure with section and subsection we would have a set of subsection fields, and also have a section field *only* if we had a section that contained some text of its own outside any subsections.

The problem with this is that it does not allow us to apply the matching scope preferences that we are seeking: there is no way of treating a match of terms `a b c` all within one subsection as any better than a match with them distributed across subsections. There is no form of field weighting with the BM25-type formula that we are working with that permits differentiation. The two less principled alternatives (assimilation to Case 1 or Case 3) just mentioned cannot allow this either, or even the superficially more plausible-looking explicit use of both section and subsection fields. For example if we give subsections a field weight of, say, 2, and sections a field weight of 1, this will still not give a better output for a match on both `a` and `b` within the narrow scope of a single subsection and a match of `a` in one subsection and of `b` in another, i.e, together only within the wide scope of the whole section.

This difficulty seems to come from the fact that where before we could have index keys occurring in more than one field, we had only one field of each type. We may now have several fields of the same type which can not be specified in advance, for instance we cannot be expected to know how paragraphs there are in a subsection.. But even if we specifically name fields as e.g. 'Section 1', or 'Background', or (de facto referring to a lower level) 'Background: early tests', this only solves the problem if we ensure that we apply the constraint, in formulating a query, that we do not permit references to field names that would allow matches on both a superordinate field and a subordinate one, for example 'Background' and 'Background: early tests'.

In the INEX Project work with XML data (Fuhr et al. 2005), queries may refer to both document structure and document content, or to content alone. Fuhr and Grossjohann (2004), applying the notion of index object to those field (element) types of interest to retrieval, compute term tfs per field, but ensure that where fields are nested there is no 'double' counting; they do not examine the impact of different field type weights. Lalmas and Rölleke (2004) examine the way in which XML-structured documents can be manipulated for retrieval purposes, focusing primarily on the form of retrieval expressions operating on hierarchically structured documents, and the way in which these can exploit content and/or structure information under tighter or looser search constraints.

This can include field weighting. Thus Lalmas and Rölleke illustrate the use of content and structure of four information types. Content information is of types Term (tf-style) and Termspace (idf-style), which may be specific for particular structure element (field) types. Structure information is of types Partof and Classpace, the former giving the value of a particular

element within a particular document (so paragraph1 may have a higher Partof value than paragraph2), the latter the general value for elements (so paragraphs may have higher Classpace values than document titles). The particular ways in which the values are set have to be independently justified, for example for Partof and Classpace by conformity with human views about potential relevance, or through machine learning. XML itself is quite neutral about indexing and matching choices: the point about XML structure is that it offers refined support mechanisms for exploiting different inputs to retrieval matching and scoring over rich hierarchically-structured documents. There are clearly many different ways in which such types of information can be combined, and in particular about the way this is done in nested structure (whether under tight or loose node specification): the principled way, implicit in Lalmas and Rölleke's illustrations, is by propagating term occurrence information upward so scores are always computed directly for higher-level elements, rather than indirectly from scores for lower-level elements; but it appears that practice in the INEX evaluation tests with XML data has varied.

Partof and Classpace are types of field weighting, giving local and global values to document elements. Thus an individual piece of text may be embedded in an upward sequence of elements, each with their own Partof and Classpace values. The implication, for Partof in particular, is that term occurrences at any level may have multiple tags representing the values of the (possibly nested) lower elements in which they figure. It is hard to see how any of this field weighting can be handled by the duplication strategy advocated in this paper: do pieces of text get duplicated e.g. at paragraph level if paragraphs are upweighted, and then again at section level if sections are upweighted? After all, sections might also contain title elements which are not themselves upweighted, compared with paragraphs. Again (assuming integer field weights), if paragraph1 is upweighted by Partof, where other paragraphs in a section are not, so we additionally duplicate paragraph1? Lalmas and Rölleke do not refer to document (element) length, somewhat surprisingly: it would presumably be another type of content information; and as Partof and Classpace are both forms of field weighting, they have implications for element length.

Liu et al. (2004) also allow for different field type weights for XML data, as one contribution to an overall term weight. Field weights are applied to field tfs, and the results are then combined with *idf*-type weights that are actually inverse *element*, i.e. field frequency weights, and also with user's own query term weights. There do not appear to be any *dl* contributions. There is an additional complication in that where fields are nested, field weights for all the nodes on the path to a text leaf are combined. In general, the technicalities of XML structure press heavily on thinking about how differential field weighting should be handled when complex hierarchies of entities are defined over base text. As a consequence, the general response to hierarchical structure is to apply subordinate score combination, as this is manifestly a convenient way of proceeding. But it is not always grounded in a clear view of what the indirect effects of the way scores are computed actually are.

Kamps et al. (2004) comment on the fact that XML documents may include elements with wildly different length ranges, far more than is common for conventional bibiographic fields; this can also occur with Web pages where, for example, text fields can range from one word long to a million words long (through spam-motivated lexical 'blow-up'). It seems in general appropriate to replace document length by element length with XML indexing, as for fields generally. However Kamps et al. point out that it is not the case that different element types (fields) are equally correlated with relevance, so the standard treatment of document length should not be simply carried over to element length. Kamps et al. explore various ways of accommodating element length in relation to potential element relevance within a language modelling framework. Analogous accommodation would thus appear to be also required within the BM25 framework, and in a way that carries over to variable field weighting. Zaragoza

(personal communication, 2005), also notes that when dealing with 'extreme' fields, like spam text, some special (and often ad hoc) treatment of element length is required, as was done in TREC04.

Language modelling may offer a way of finessing the overlapping fields problem because it makes it easy to build separate models for any set or subset of items, whether different whole document sets or different field types or, as in Sigurbjörnsson et al. (2005), different element types in hierarchically structured XML documents. The presumption, even where such strongly overlapping fields as nested elements are concerned, is that the various language models can be treated as if they were quite independent, as the approach adopted in Sigurbjörnsson et al. (2005) implies. However it is not clear that their probabilistic function for scoring matches using multiple models is really different from the obvious score combination adopted in Metzler and Croft (2004) and hence, because it too relies on term frequency information, does not also suffer from the same problems of score combination as under field weighting. Thus it is necessary even with apparently straightforward uses of language modelling, to ensure that matching functions behave properly under field weighting. There is also the problem of sufficient training data, especially for Classpace and, even more, Partof in Lalmas and Rölleke's case.

Passages

Some similar difficulties arise if we operate with passages instead of conventional section structure, and these are defined by fixed windows with offsets (as for continuous transcribed speech data in Johnson et al. 2001): thus suppose we use 100-word windows at 25-word offsets: this will naturally produce 4-deep overlaps for each 25-word stretch and hence automatic key repeats at the passage level. There is of course no problem if passages, whether disjoint or overlapping, are used simply as a single field type, as the basis for ranking their parent documents rather than as retrieval objects in their own right, and document matching scores are just those for the best matching passage alone. The problem is dealing with multiple contributors to scores. Thus Kwok (2005) simply combines the scores for multiple (disjoint) components (e.g. paragraphs or fixed windows) to obtain an overall document score, though the argument presented in CIKM04 suggests that this is improper when the component score depend on $tf * idf$-type weighting. Cormack et al. (2005) use multiple overlapping passages with complex length-related weighting generating a combined final document score, but this does not involve $tf * idf$ contributions.

Field labels

Once we consider field labels themselves a sources of information (as opposed to tags for what is below them), we have to handle them as independent fields according to whatever Case model is appropriate. Field type indicators, like 'section', may not often justify such an independent status, though we can imagine they sometimes could: for example a markup language that used the indicator 'Example' could be directly useful for searching. This is rather similar to the content heading 'Background' or 'Experimental data' which could also be useful: if we cannot rely on documents having a field called something as specific as 'Background' we can search the section label field for the word 'Background' just as we may search a title field for some word; and since in general section headings can range over a whole vocabulary, this is what we would want to do, in Case 1 style.

The XML protocol would always assume that anything like a section title has explicit field status independent of the dominating node 'section' which is thus viewed formally, even if in other cases a section node can immediately subsume leaf-level text. But maintaining such clear divisions may not always be easy, as "example" suggests. This might be treated as formal structural node label, like 'section', rather than as a node title, but this would make it impossible

to search for the index term "example" under the presumption that it can simply match either structural labels or within leaf text: a more explicit heavy-duty sear h specification would be required. The treatment of node labels is, however, independent of the issue of dealing with hierarchical structure.

However when we do this we are only implicitly, just as in the non-hierarchical situations discussed earlier, using any hierarchical structure: the issue is using hierarchy directly.

Hierarchical structure is clearly a challenge for the whole approach we have considered: we would like to be able to apply BM25 to it the information that document structure conveys. Current work on XML suggests some possible routes to explore, but the right way of using the CIKM04 ideas (and of practically implementing them) in such contexts needs much further investigation.

5 Queries

In CIKM04, queries are assumed to be simple unitary items, for example the set of terms `[ a b c ]`: the question addressed there is, given such a set of query terms, can we handle them correctly in relation to documents with different fields.

But one consequence of extending the range of retrieval situations to cover the document Cases 2 and 3 is that we have to allow for query-document field correspondences or *attachments*: thus if we take a document field with a distinct term vocabulary, for example (let us suppose), own-URL, this implies that there is also a query field 'own-URL'. This may be explicitly flagged, or is implicitly flagged because it is interpreted in a field-specific way in matching. As a practical matter, with the former the user knows what is going on, with the latter perhaps not, but the logic from the retrieval system point of view is the same. In other words, just as documents have key types, so have queries their corresponding types.

Thus given a simple CIKM04 query with no apparent structure, if documents have the field title, abstract and text, say, the query is to be interpreted as having a corresponding structure, and the query terms are all assigned to each field. It is not necessary to show this structure explicitly, but it is logically there, captured in the index and match operations on the documents.

A query can of course specify any document field: where terms can occur in more than one field this has to be done explicitly, with document key markers it can be done implicitly. Thus where terms can occur in, for example title and abstract, the query might specify a match on title only. The XML query language XIRQL (Fuhr and Grossjohann 2004) is designed to handle this situation correctly given the potential complexity of XML structure representation and tree access paths.

We have also to consider whether to allow query-specific field weights for corresponding fields. If the fields are differently weighted in the documents, they are automatically differently weighted for the query. On the other hand, if we suppose the user wants to give additional emphasis to some fields, beyond what the system has as defaults (assuming the user knows what these are), there would seem to be no problem about doing this, applying the CIKM04 mechanism. We already allow for query term weights greater than 1 within the framework of BM25-based matching, so if we have a user who sees the title field as extra important, we can increase the tfs for the title terms by replicating the *query* title field. The effect would be to enhance a query-document match beyond what the system normally offers. As noted earlier Liu et al. (2004) allow users to upweight preferred fields, i.e. allow 'ad hoc' query-specific document field weighting rather than permanent field weighting, but within a different scoring framework.

Suppose, on the other hand, that a user knows that the system upweights title term occurrences and doesn't like this: it would presumably be possible - again assuming the user knows

what the system's title field weighting is - to downweight the query title field and so reduce query terms tfs to the required compensating extent. If the user only specifies a single field, it does not matter what the document field weight is: the effect on the output ranking will be the same as with no field weight.

There is a different situation where we have queries with their own fields which are not correlated with document ones, as illustrated by TREC topics. These have their own titles (call them qtitles), descriptions, and narratives. We can clearly assign these query fields different weights by CIKM04-style replication. We have higher tfs for these terms which will then combine with the information for whatever fields are matched, and field values applied, for the documents. Thus we might rank a document that matched the query description field terms with document title ones extra highly,

There are other types of query structure that the approach described here cannot handle. This applies, obviously, to pure Boolean queries of the conventional kind, since the BM25-style retrieval model with term weighting is designed to overcome the limitations of the Boolean approach. However once Boolean is treated as a special case of weighted searching, as in Salton et al. (1985), the strategy discussed here naturally applies. There are also many familiar versions of proximity-based searching. The earlier discussion of hierarchical document structure showed there is no natural way, within the present simple replication framework, of handling any proximity-based query that uses query structure itself, dynamically, to define preferences over scopes, as opposed to preferring fields that are de facto indicators of scope, as with preferring a title field match over a (remaining) text one. There is also no way of handling what might be thought of as variable proximity within a single field, for example giving a higher weight to a joint match for 2 terms within 3 words than to a joint match within 6 words: logically speaking this is the same situation as where we have explicit fields: here we have two mini-fields, one including the other. Thus as mentioned earlier, the correct treatment of such types of query-specified logical fields has to be done when term weights are computed from their components, with appropriate adjustment of the component values (which may be practically extremely tricky). Just requiring a match within some proximity window, on the other hand, is a variation on Boolean conditioning, and is thus outside the BM25 approach. The MultiText strategy discrobed in Cormack et al. (2005) uses multiple subqueries with complex relationships to one another, but in essentially Boolean style without term weighting.

We may also, as mentioned in the Introduction, have multiple whole query representations, as considered in Turtle and Croft (1990), Belkin et al. (1995), Croft (2000) and Metzler and Croft (2004). The idea is that, just as we originally recognised that we could think of document fields, or representations, or streams, as reflecting different views of the underlying document content, we may also have different interpretations or views of the underlying user's information need. We can in fact treat the structure of TREC topics, as described above, in this more abstract way. But Belkin et al. approach alternative query representations more explicitly, by taking different formulations of the query by different human searchers and applying them together on the basis that this will give better performance than any one alone. This *query combination* is treated as evidence combination within the Turtle and Croft network model. Belkin et al. also treat the different ways that systems as wholes derive search queries from an initial user request as providing different query representations but here, since the systems may operate in quite different ways, reach their final search output by a *data fusion* process applied to the set of ranked outputs.

Croft, and Metzler and Croft, treat these two cases, and also retrieval from multiple databases, within the overall abstract framework of evidence combination. As noted earlier, this allows for very different sorts of document key, for example terms, phrases, citations, links, controlled index headings. Some of these correspond to quite distinct document fields, others to logically different

fields ranging over the same initial document manifestation (notably the body text). Different representation types may be combined in (arbitrarily complex) ways in structured queries: in some cases these are best seen as projecting keys of some types, like proximity phrases, back into neutral document representations of the form [term+text location].

Documents in this approach have individual feature representations (e.g. have particular terms), and these must be typed for matching purposes, but there is no simple way of referring directly to fields a wholes. It would presumably be possible to have a representation type layer, which could have its generic weights, whether arbitrarily assigned or learnt, propagating down to individual representation keys. However the primary means of upweighting key types is via their appearance in the query: Metzler and Croft refer to both automatic and manual type weighting. Then since document representation nodes already have weights similar to the BM25 ones, combining tf, *idf* and *dl*, and documents are scored by combining matching query representations according to operator rules, this implies that key type weighting has the defect that CIKM04 was designed to overcome.

6 Discussion

CIKM04 shows that the decompositional treatment of tfs is the right way to treat differential field weights. But with the simple (Case 1) situation considered there, there is a loss of information: any meaning in the field labels themselves is lost and also any meaning in the field relationships, though there is information in labels like "title" and "abstract" and hence value in having that information available for the user, and there is information in relationships, for example in the hierarchical case that this material is part of that material.

With the strategy described in CIKM04 (and TREC04) there is no way for the user to indicate an interest in titles, since this cannot be mapped into any document data or area; equally, there is no way that what constitutes title information for the document can be given to the user, unless there is a completely separate display version of the document. This is indeed likely to be the case, but can only be used indirectly by the user for searching, e.g. by picking out new query terms to submit. It might be argued that label information can be retained by including labels as keys, but this is not necessarily pointful in itself, as indexing with the *word* "section" suggests, and in any case such keys would not carry relative locality information to exploit.

However, consideration of other cases than the simplest one discussed in CIKM04 shows that there is nothing in principle in the strategy that prevents the retention of field ties, i.e. index key type information, for terms: even when we have a document to which both Case 1 and Case 2 apply (and we can take Case 4 either way), and simply throw all the terms into one big pot, there can be a good reason to attach field flags to terms, namely to ensure that ambiguous symbols are treated correctly; and given that data, there is no reason to prevent users from specifying matches only on particular fields. The point of this paper is rather to explore the consequences for document scoring of allowing matches to be made on several fields, especially when these may be differently weighted.

Thus when we consider a wider range of situations than those addressed in CIKM04, it is evident that the CIKM04 strategy can be extended, and in a systematic way, to more complex key type and key situations. However as the earlier reference to the challenges presented by field-dependent *idf* suggests, even when working within the most basic BM25 framework as far as index key types are concerned, it is not obvious how information geared to different document locations should be combined. More generally, there comes a point where matching scores rather than index data have to be combined. But even in these complex cases, the data decompositional approach presented in CIKM04 is a useful analytical tool. It is useful, in particular, as

a way of studying some implications of field weighting in the context, on the one hand, of the strategies for handling multiple document representations exploited in the INQUERY approach to retrieval, and on the other hand, of the rich information about document structure, and especially hierarchical structure, that XML makes available. It also offers another route into thinking about the distribution of information between document and query, which in some cases is only a matter of operational convenience but in others is a fundamental feature of retrieval itself. Thus just as, earlier, the differences between file time indexing and search time indexing turned out to be more subtle than initially appreciated, so there are complexities about the way data pertinent to index key weighting are distributed between document and query representations and over successive steps in computing matching scores.

References

Belkin, N. et al., 'Combining the evidence of multiple query representations for information retrieval', *Information Processing and Management*, 31, 1995, 431-448.

CIKM04 - see Robertson et al. 2004.

Cormack, G.V. et al., 'MultiText experiments for TREC', in *TREC: Experiment and evaluation in information retrieval*, Ed. E.M. Voorhees and D.K. Harman, Cambridge, MA: MIT Press, 2005, 347-371.

Crestani, F., Vegas, J. and de la Fuente, P., 'A graphical user interface for the retroeval of hierarchically structured documents', *Information Processing and Management*, 40, 2004, 269-289.

Croft, W.B., Turtle, H.R. and Lewis, D.D., 'The use of phrases and structured queries in information retrieval', *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval (SIGIR '91)*, 1991, 32-45.

Croft, W.B. and Turtle, H.R., 'Retrieval of complex objects', *Proceedings of the 3rd International Conference on Extending Database Technology (EDBT)*, Berlin: Springer, 1992, 217-229.

Croft, W.B., 'Combining approaches to information retrieval', in *Advances in information retrieval*, Ed. W.B.Croft, Dordrecht: Kluwer, 2000, 1-36.

Frommholz, I., Applying the annotation view on messages for discussion search', *Text REtrieval Conference (TREC 2005) Workbook*, National Institute of Standards and Technology, Gaithersburg MD, 2005, 218-227.

Fuhr, N. and Grossjohann, K., 'XIRQL: an XML query language based on information retrieval concepts', *ACM Transactions on Information Systems*, 22, 2004, 313-356.

Fuhr, H. et al. (Eds.), *Advances in XML Information Retrieval. Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, Berlin: Springer, 2005.

Hawking, D. and Craswell, N. 'The very large collection and web tracks', in *TREC: Experiment and evaluation in information retrieval*, Ed. E.M. Voorhees and D.K. Harman, Cambridge, MA: MIT Press, 2005, 199-231.

Johnson, S.E. et al., 'Spoken document retrieval for TREC-9 at Cambridge University', *The Ninth Text Retrieval Conference (TREC-9)*, Special Publication 500-249, National Institute of Standards and Technology, Gaithersburg MD, 2001, 117-126.

Jones, G.J.F., 'Exploring the incorporation of acoustic information into term weights for spoken document retrieval', *Proceedings of the BCS-IRSG 22nd Annual Colloquium on Information Retrieval Research*, 2000, 118-131.

Jones, G.J.F. et al., 'Retrieving spoken documents by combining multiple index sources', *Proceedings of the Nineteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval (SIGIR '96)*, 1996, 30-38.

Kamps, J., de Rijke, M. and Sigurbjörnsson, 'Length normalisation in XML retrieval', *Proceedings of the Twenty Seveenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004)*, 2004, 80-87.

Keen, E.M., 'Term position ranking: some new test results', *Proceedings of the Fifteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval (SIGIR '92)*, 1992, 66-76.

Kwok, K.L., 'Ad hoc retrieval at TREC using PIRCS', in *TREC: Experiment and evaluation in information retrieval*, Ed. E.M. Voorhees and D.K. Harman, Cambridge, MA: MIT Press, 2005, 321-345.

Lalmas, M. and Rölleke, T., 'Modelling vague content and structure querying in XML retrieval in a probabilistic object-oriented framework', *Proceedings of the 6th Conference on Flexible Query Answering Systems*, Lyon, April 2004.

Liu, S., Zou, Q. and Chu, W., 'Configurable indexing and ranking for XML information retrieval', *Proceedings of the Twenty-Seventh Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04)*, 2004, 88-95.

Metzler, D. and Croft, W.B., 'Combining the language model and inference network approaches to retrieval', *Information Processing and Management*, 40, 2004, 735-750.

Mitra, M. et al., 'An analysis of statistical and syntactic phrases', *Proceedings of RIAO 97, Computer-Assisted Searching on the Internet*, Montreal, 1997, 200-214.

Robertson, S., Zaragoza, H. and Taylor, M., 'Simple BM25 extension to multiple weighted fields', *Proceedins of the 13th ACM Conference on Information and Knowledge Management*, 2004.

Salton, G., Fox, E.A. and Voorhees, E., 'Advanced feedback methods in information retrieval', *Journal of the American Society for Information Science*, 36, 1985, 200-210.

Sigurbjörnsson, H., Kamps, J. and de Rijke, M., 'Mixture models, overlap, and structural hints in XML element retrieval', in *Advances in XML information retrieval (INEX 2004)*, Ed. N. Fuhr et al. Berlin: Springer LNCS 3493, 2005.

Silveira, M. and Ribiero-Neto, B., 'Concept-based ranking: a case study for the juridical domain', *Information Procesing and Management*, 40, 2004, 791-805

Skounakis, M. and Craven, M., 'Evidence combination in biomedical natural-language processing' *Proceedings of the 3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics (BIOKDD03)*, 2003, 25-32.

Spärck Jones, K., Walker, S. and Robertson, S.E., 'A probabilistc model of information retrieval: development and comparative experiments', Parts 1 and 2, *Information Processing and Management*, 36, 2000, 779-808 and 809-840.

Strzalkowski, T. et al., 'Evaluating natural language processing techniques in information retrieval', in *Natural language information retrieval*, Ed. T. Strzalkowski, Dordrecht: Kluwer, 1999, 113-145.

TREC04 - see Zaragoza et al. 2004.

Tsikrika, T. and Lalmas, M., 'Combining evidence for Web retrieval using the inference network model: an experimental study', *Information Processing and Management*, 40, 2004, 751-772.

Turtle, H.R. and Croft, W.B., 'Inference networks for information retrieval', *Proceedings of the Thirteenth International ACM/SIGIR Conference on Research and Development in Information Retrieval (SIGIR '90)*, 1990, 1-24.

Turtle, H. and Croft, W.B., 'Evaluation of an inference network-based retrieval model', *ACM Transactions on Information Systems*, 9, 1991, 187-222.

Zaragoza, H, et al, 'Microsoft Cambridge at TREC-13: Web and HARD tracks', *The Thirteenth Text REtrieval Conference (TREC 2004)* (2004), SP 500-261, National Institute of Standards and Technology, Gaithersburg MD, published electronically via http://trec.nist.gov