Language modelling and relevance

Version 4.32 14 June 02

Karen Spärck Jones
University of Cambridge
Cambridge, UK
ksj@cl.cam.ac.uk

Stephen Robertson
Hugo Zaragoza
Microsoft Research
Cambridge, UK
{ser,hugoz}@microsoft.com

Djoerd Hiemstra
University of Twente
Enschede,
The Netherlands
hiemstra@cs.utwente.nl

This paper in its final form appeared in *Language modelling for information retrieval*, ed. W.B. Croft and J. Lafferty, Dordrecht: Kluwer, 57-71

1 Introduction

This paper addresses three questions about the Language Modelling (LM) approach to information retrieval. These questions are about LM and relevance. They arise because relevance has always been taken as fundamental to information retrieval (see, e.g. Saracevic [10] or Mizzaro [7]). Thus from the standpoint of retrieval theory, the presumption has been that as relevance is the key notion in retrieval (for how could it not be?), this should be explicitly recognised in any formal model of retrieval. The Probabilistic Model (PM) of retrieval does this very clearly. Turtle and Croft [13, 1] present the INQUERY network model as a probabilistic classifier testing for whether the hypothesis of relevance holds given the evidence supplied by document and query. The Vector Space Model assumes that some of the points in the multidimensional information space of a retrieval system represent relevant documents. But the LM account of what retrieval is about seems quite different: relevance does not formally figure in it at all.

A retrieval model that does not mention relevance appears paradoxical. But the form in which the LM account is expressed immediately provokes the question:

1. What about relevance?

and as a natural consequence, in the face of retrieval realities:

2. What about multiple relevant documents?

and hence, for the design of retrieval systems:

3. What about relevance feedback?

In this paper, we consider these questions in more detail. We examine the status of relevance itself in the LM view, and its implications for multiple relevant documents and relevance feedback. We present the problems we see and outline some possible solutions to them. The paper generally takes an informal approach; little mathematical detail is presented. The aim is to explore the concepts behind the models rather than the mechanics of modelling.

For our present purpose we will treat LM as developed by Croft and his colleagues at the University of Massachusetts, Amherst and later at Mitre Corporation, by Hiemstra and others at University of Twente, by Lafferty and others at Carnegie Mellon University, and at Bolt, Beranek and Newman (see references under Section 2), as representing a single theory, unless specific distinctions are required. We will develop our arguments by contrasting Language Modelling with Probabilistic Modelling (PM) as formulated by Robertson and his colleagues, taking [11] as a convenient default reference point for this. Moreover, though LM is well established for speech processing and has been explored for translation and summarising, for instance, these other tasks – with the possible exception of topic detection and tracking (see Walls et al. [14]) – do not have the particular properties of document/text retrieval on which we wish to focus, and we will therefore confine ourselves to the retrieval case.

2 Relevance in LM

2.1 The status of relevance

LM approaches the relation between a document and a request by asking: how probable is it that this document generated the request? More strictly the question is: how probable is it that the document, as represented by its index description, generated the request as represented by its indexing description, i.e. the search query? There is no explicit reference to relevance here. However the presumption is that if it is highly probable that the document generated the request, then the document's content is relevant to the information need underlying the user's request. A good match on index keys implies relevance, though relevance is never mentioned in the model.

PM asks a quite different question: how probable is it that the document is relevant to the request? More properly the question is: how probable is it that the document content is relevant to the user's need? Then since both content and need are unobservables, it is necessary to provide an argument for taking index descriptions as levers in deriving the operational equivalent of probability of relevance. The PM approach thus leads to the same position as the LM one: a good match on index keys implies relevance. However, in contrast to the LM approach, relevance figures explicitly in the PM model.

This stems from the fact that, strictly, LM assumes there is just one document, the 'source' document, that generates the request and that the user knows (or correctly guesses) something about this document. Even if the system does not deliver this document first, the user recognises it as the generating document when they see it. One view of this account is that as there is only one document that matters in the file, whether it is called a generating document or a relevant document is just a matter of labelling.[1]

[1]We are aware that Lafferty and Zhai, in a paper in the present volume, argue that no such assumption is required for the usual language model for IR. We believe that the matter is not yet fully resolved.

This does not seem wholly satisfactory. However the only way of saving relevance as an independent notion within the source document framework is to say that the user has an ideal document in mind, i.e. some notional relevant document, and judges a particular document (the erstwhile generating document) as relevant if it comes close to this ideal (see Ponte [9] and Miller et al. [6]). But this compromises a key LM advantage. The LM approach is quite explicit about the form of the measure of association or similarity between document and request. LM, like PM, eschews the vague notion of similarity that characterises many other approaches and works with a measure that follows from the model assumptions. In the LM case this applies only if the assumption is made that there is a specific precursor document, i.e. the measure of similarity is the probability that a particular document is the generator. Invoking a notional ideal document implies another, and weaker, notion of similarity between a specific document and this ideal one.

But does this difference between the formal models matter when their operational interpretations, and hence manifestation for the user, are the same? The LM approach might seem to have the advantage that it does not deal directly in unobservables, since even if it allows for different forms of index description these are still derived from observable initial documents and requests. However a more honest view is to accept that LM is being applied to retrieval, which deals in content and need, so this surface distinction between LM and PM is somewhat spurious. It is possible to *apply* LM as an effective retrieval tool, without bothering about the prior reasoning which led to its design, much as one can find that one saw cuts wood rather better than another without having to know what the functional explanation for this is. Thus LM can be seen (and has been used) simply as a good way of scoring document-query matches and hence of organising system output. But as the analogy with saws suggests, such an uncritical use of a tool may have its dangers: anyone using a power saw for the first time is well advised to read the instruction book, and especially the safety page.

2.2 Multiple relevant documents

In all ordinary experience of retrieval, it is a fact of life that there may be more than one relevant document for a request; indeed there are normally several, and with large files there may be hundreds.

Formally, LM takes each document D and, using its individual model $P(T_i|D)$ in conjunction/comparison with a generic model $P(T_i)$ (the file model), asks how likely it is that this document generated the request, by assuming that query terms T_i are conditionally independent given the document. The basic formula used in several of the papers which take a language modelling approach to IR can be written as follows:

$$P(T_1, \cdots, T_n|D) = \prod_{i=1}^{n}((1-\lambda)P(T_i) + \lambda P(T_i|D)) \tag{1}$$

The outcome is different values for documents. This conveniently leads to a ranking, and for practical purposes the comparison output is treated like that for any other retrieval method that induces a ranking, i.e. as delivering multiple relevant documents if these exist. System performance can also be measured using rank-based methods. However as just mentioned, the principle underlying the model is that it is identifying *the* document that generated *the* request. This document ought to be the one with the highest value, but since the base for estimating value is not perfect, the generating (i.e. relevant) document could be another one.

However once this relevant document is recovered, i.e. is encountered going down the ranking, retrieval stops.

The way the LM approach is applied in practice, that is, subtly changes its meaning. As long as documents are really different, they should generate different requests. The actual request may not be the best they could generate (indeed, will obviously not be), but that's tough and incidental. Similarly if the document index descriptions used for generation are impoverished, e.g. consist of two or three subject headings as opposed to full text, several documents could be deemed equally likely to have generated a given request. Or if the indexing language is restricted, different requests could be represented by the same search query. But this again simply implies that the base for determining the real relative status of different documents vis-a-vis a request is weak, not that there are no real differences of status.

Thus if we now consider the situation where we recognise the empirical fact that there may be several documents relevant to the user's information need, and where we also have some particular expression of that need as a request, what does this imply for the formal LM account of retrieval?

We exclude, for the moment, the use of feedback. We assume batch searching (and also that whatever means the user has exploited to develop their request are outside the retrieval system proper). Then the LM approach most obviously works as follows. We determine how probable it is that each document generated the request. Then since each relevant document has to be viewed as independent of every other and, further, in assuming that this document generated the request we have formally to treat all the other relevant documents as non-relevant, we notionally conduct a series of R independent searches for each relevant document D_j $(1 \leq j \leq R)$ alone. The logic is the same as if we actually had several needs each with its own single relevant document and its own request, but the requests just incidentally happen to be worded in the same way. Then we might seek to find the set of documents which maximises the probability:

$$P(T_1, \cdots, T_n | D_1, \cdots, D_R) = \prod_{j=1}^{R} \prod_{i=1}^{n} ((1-\lambda)P(T_i) + \lambda P(T_i|D_j)) \tag{2}$$

In practice, we might economise and simplify the set selection process, and emerge with the same single ranking of documents as in conventional systems, but this would be vulgar practice, not proper principle.

Unfortunately, this way of saving the theory is not convincing as a way of capturing empirical facts: real users can, of their own accord, accept several documents as relevant to their need. At least, more fieldwork is needed to establish that they are always *really* working with a family of needs and, more especially, with a sequence of needs when accepting documents while assessing a sequence of documents.

However if we try to incorporate the reality that there may be several documents relevant to the need into the model, this formally implies not that we look for the single document that generated the request, but that we look for the *set* of documents that generated the request. That is, the relevant document set is the set of documents that most probably generated the request. But of course this way madness lies, or at least we have a large combinatorial challenge.

Formally, we have the exact same situation as in Equation 1 considering the set of documents $\tilde{d}$, but here the relevant subset of documents is acting as a hidden variable. As such, it

can be disposed of by marginalising it, at the cost of having to consider all its possible values, in this case, all possible subsets of the document collection, denoted as D^*.

$$P(T_1, \cdots, T_n|D) = \sum_{\tilde{d} \in D^*} P(T_1, \cdots, T_n|\tilde{D} = \tilde{d}) P(\tilde{D} = \tilde{d}|D) \tag{3}$$

This might not be felt to be too much of a problem, given modern machines and some smart mathematical analysis. But there is a more important difficulty about the strategy. This is that given that the information available to determine the set is only the initial request, the chance that the most probable set will coincide with the relevant set is rather low.

An interesting way out of the multiple relevant documents problem is to reverse the language models: instead of assuming that queries are generated from documents, we assume that documents are generated from a request model, or perhaps more generally, from a *relevance model*. This approach has been adopted by some authors for the topic detection and tracking task [3, 12], where the request actually consists of a known relevant document. However, some LM authors working with the traditional IR task have also attempted to develop relevance models.

Lavrenko and Croft [5], who start with the aim of combining ideas from PM and LM, consider a model in which the query terms are generated from a mixture of single-document models. Lafferty and Zhai [4] consider not only document models, but also a language model based on the request, first a simple one involving query terms only, and then a more complex one in which the user is assumed to choose the query terms from a succession of documents found in the course of a search. However, in both cases they find it necessary to make an assumption similar to that mentioned earlier involving an idealised relevant document, about some user measure of similarity (not specified by the model) between the relevance model and the document models of the relevant documents.

Both approaches lead to a two-stage retrieval process which is very much like so-called pseudo-relevance feedback. In the first stage, a query term is used to rank the documents (just as in 'normal retrieval'). In the second stage, the probabilities of the top ranked documents are used to infer a relevance model, and the documents are ranked by the probability that they were generated by this relevance model. Lafferty and Zhai also indicate that their approach could be used for real relevance feedback, though they do not test this possibility.

Clearly, the multiple relevant document problem is of particular concern if we wish to make use of real relevance feedback. Thus there are good reasons for asking how relevance feedback fits the LM approach. As the LM approach, like the PM one, benefits from having more information for estimating probabilities, it is rational to ask how knowing about some relevant documents can improve estimation, and hence retrieval, for others.

2.3 Relevance feedback

In the pure LM approach, with a single generating document, there is of course no motivation for relevance feedback proper. Indeed while the probabilities assigned to all the documents in the collection should change, in an appropriately consistent way, as the user inspects proffered documents, once the user has found the generating document, the only but otiose form that feedback could take would be to be negative. What, however, are the implications for the LM approach if we try to accommodate multiple relevant documents?

Suppose that we do our first search and find our first relevant document, i.e. the document that generated our initial request. If we modify the request, say by adding terms taken from

the document, the LM approach strictly requires that we now treat this as a new request and find the document that most probably generated it. But of course since the new request has been explicitly built from a document - in the limiting case we might just have adopted that document's description as the new request - we have to withhold that document in the search or we will simply go round in circles. We thus change the file (if only infinitesimally) as well as the request, i.e. change the component of the situation that the LM assumes will remain the same: even if the status of the documents in the file alters, the presumption is that the membership of the file is constant over the user's search session.

It is nevertheless clearly possible to continue on the one by one basis. But it has to be accepted that if there is no reference to an underlying need and a process that is intended to improve the expression of that need so as to increase the chance of identifying all the documents relevant to it, the LM approach is dealing with a succession of needs and corresponding requests. The real life analogue is routing rather than adhoc retrieval. Moreover, the one-by-one model of retrieval that is entrenched in the LM approach naturally implies a step-wise modification (i.e. replacement) of the last version of the request. This does not necessarily lead to the same eventual request for retrieving the last relevant document as when requests are modified (or replaced) using the information supplied by a set of relevant documents all taken together.

Hiemstra's [2] proposed solution to the multiple relevant documents problem is to assume that each relevant document independently generated the request as stated in Equation 2, and to give the standard LM approach a slightly different justification. The usual justification for Equation 2 is a form of 'smoothing': the model for a specific document is smoothed by the generic model for the collection as a whole in order to avoid certain extremes such as zero probabilities assigned to some terms. Hiemstra's basic assumption is the same (the user is assumed to have a specific document in mind and a request and then search query is generated on the basis of this document); but instead of smoothing, the generation process is assumed to assign a binary importance value to each term position in the query. The position for an important term is filled with a term from the document; a non-important one is filled with a general language term. If we define $\lambda_i = P(\text{term position } i \text{ is important})$, we get:

$$P(T_1, \cdots, T_n|D) = \prod_{i=1}^{n}((1-\lambda_i)P(T_i) + \lambda_i P(T_i|D)) \tag{4}$$

Then in feedback each known relevant document provides evidence for the importance of each query term, and the evidence across the known documents is accumulated to provide estimates of importance for the different query terms. However, this strategy assumes that each query term has a (general or average) importance status which is independent of particular documents, and this is incompatible with the generation process on which the LM approach is founded. For example, if a term does not occur in a document it must have been *un*important when the request was generated, but may nevertheless now be deemed important. This clearly subverts the whole LM idea.

Relevance feedback in general allows for either or both of query reweighting and query expansion. Hiemstra's strategy is only for reweighting. Ponte [9] addresses expansion. He combines LM information with other more conventional data about term value in order to produce an enlarged query. This is treated as a new query for which the generating probabilities for the file documents that have not already been inspected are computed. But in fact, the rationale for the new part of the query lies in the set of documents already seen,

not in the unseen putative generating set, while at the same time the values for the old part have already been computed for all the documents in the file and exploited as the base for the user's past assessments. This is not a consistent, principled interpretation of LM for relevance feedback.

Neither Hiemstra nor Ponte, therefore, provide a fully convincing way of handling feedback within the LM framework, and specifically a proper account of model training on relevance data. It may be that such an account can be found in the more recent models of Lafferty and Zhai [4] and Lavrenko and Croft [5], discussed in the previous section, though they have not been developed in this direction.

Overall, LM in its classical form emerges as a minimalist theory about the information that can be drawn from multiple relevant documents, that they supply about one another, and hence can jointly supply to leverage a query that best retrieves them all. PM responds to this situation in a more wholehearted way.

3 A possible LM approach: parsimonious models

What would a proper LM approach to the multiple-relevant-documents case look like?

One suggestion is that there should be an explicit model which generates the *set* of relevant documents, i.e. that relevance sets should be built into the model from the start rather than in the 'brute force' fashion sketched in 2.2 above. In this way relevant documents could be seen in the LM approach to have some similarity, different from the whole collection, which would in principle allow some form of relevance feedback. But such a model would seem to have considerable difficulties.

The LM approach at present allows for $N + 1$ language models, where N is the collection size. The additional one is the general language model. The relationship between this last and the individual document models does not seem to have been examined for the aggressively generative theoretical stance on LM for retrieval being examined here. Thus how can a document be generated from one language model when the entire collection is generated from a different one? There is an issue here for LM in itself, but there is clearly much more of a problem when we start looking at models for sets of relevant documents.

Is there any mechanism within the LM field to support some kind of global-and-local model structure for the sets case? One possible mechanism is the one with which we are familiar: the mixture model. However, the mixture model implies a curious view of the interaction between the various parts – *two* distinct (in this case unigram) language models with unrelated probabilities for the same event, *and* a combination model for deciding which of these to use.

This view would seem to bear little relation to any real authorship process. The language that we use in writing a paper on a specialist subject is basically the same language we use when talking to our families, with only some parts modified by the particular contexts. It seems that what we need is a general model for some large accumulation of text, which is modified (not replaced) by a local model for some smaller part of the same corpus. This is resorting to a perfectly conventional account of language use as underspecified or highly abstracted in the large, but involving specific form (and content) variations in text type (register, etc.) in any individual situation. In the present case, where we are dealing with simple index terms, this local variation is in the uses of or selections from the language vocabulary.

But having just a two-level account, covering a whole-collection model and a single local model, is not enough when we have a set of relevant documents rather than a single one. Although we are considering documents that are all relevant to the same one request, this cannot be taken to imply, unrealistically, that these documents are identical, or are generated by a single model. We have to allow for the fact that requests as topic specifiers can be selective on the normally much richer topic content of documents, especially when the necessary variation in topic granularity is factored in. Even if the presumption is that (the whole of) a (highly) relevant document is *about* the request topic, this does not imply that the document has just one topic. Documents are multi-topic, can be relevant to more than one different request, and can resemble other different documents in different ways.

Then if we accept, as we have to, that documents are multi-topic, and that relevance to a request topic in the retrieval sense is only one topic relationship for a multi-topic document, we actually need not a two-level but a three-level LM structure:

1. A whole-collection or generic model...
2. ...modified by a relevant-documents model...
3. ...modified by an individual document model.

The third model includes all the special aspects of a particular document that are specific to topics other than the topic of this particular user request. It would play something like the role of the error or residual term in a regression or ANOVA model – "all the variance not explained by the above". All of these distinct models appear to be necessary for a proper application of LM in the retrieval context without making the unsatisfactory assumption that all relevant documents are the same.

Before describing in a little more detail in section 3.2 how such a system of models might be applied in the retrieval context, we need to discuss the notion of *parsimonious* models. This is a necessary feature of a structure with models at more than one level.

In what follows, the concepts of 'explanation', 'generation' and 'prediction' are taken as more or less equivalent: saying a model *explains* a piece of text is the same as saying it *generates* it (in the language model sense), or that it *predicts* it.

3.1 The need for parsimony

One difficulty with the multi-level approach suggested above lies in comparing different models for their explanatory power concerning a piece of text. If a model can incorporate as many parameters as it wants, then it can fit any fixed set of data to any degree of accuracy. In principle, a document-specific model can explain a document completely (or rather, explain it within the limits of the basic LM approach, e.g. bigram, being applied), and has no need to appeal to a higher-level model of a collection of documents. The only sense in which such a higher-level model might contribute is in the sense of explaining the data more parsimoniously. Thus we may have a generic collection model to cover all the documents taken together, along with individual document models that need only a few parameters to distinguish each document from the others. So although there are two levels of model, their combined complexity is less than that implied by having a complete set of full individual document models all on the same level.

For example, consider the usual LM approach, represented by $[\lambda P(T|D) + (1 - \lambda)\, P(T)]$ (Equation 2). This formula defines a mixture of a generic collection model and a specific

document model, controlled by the parameter λ. From a maximum likelihood perspective, and without any constraints on parsimony, $P(T|D)$ is always preferable to $P(T)$, since it will best explain the observable data. From this perspective λ should be close to 1, so (unfortunately) the least parsimonious explanation is obtained.

On the other hand, parsimony would clearly prefer a probability estimate for T similar to $P(T)$ for most terms, that is: $[\lambda P(T|D) + (1 - \lambda)\, P(T)] \approx P(T)$. We see that this is only possible if: i) $\lambda \approx 0$, in which case all words have the same probability of emission regardless of the document they belong to, or ii) $P(T|D) \approx P(T)$, that is the term T appears in the document with exactly the same distribution as it appears in the corpus, which would certainly not apply to the terms and documents we are particularly interested in.

Since λ is independent of T in the LM approach of Equation 1, we could not 'zoom-in' or 'explain-away' the behaviour of certain terms only under this framework – we would have to do so *for all terms* in the same degree. What we would need for parsimony is for λ to be equal to zero *for most terms and for most documents*, similar to the separate λ for each query term suggested by Equation 4. But in this case λ would have to depend on T and D. We would need to select or *fit* lambda values for each term in each document.

So, for example, a generic unigram model for the collection as a whole, accompanied at some lower level by a small number of specific unigram models, each of which had parameters for only a small number of terms with distinctive document behaviour, would be a very parsimonious model for a collection of documents. These lower-level models might be at the level of individual documents, or of individual topics, for example. For this whole strategy to work, the way any of these models, whether individually or in combination, are fitted must reward parsimony. There are approaches to statistical model fitting, particularly some Bayesian and maximum entropy ones, which do reward parsimony. But as far as we know no such model has been used in connection with LM and retrieval.

3.2 A 3-level model again

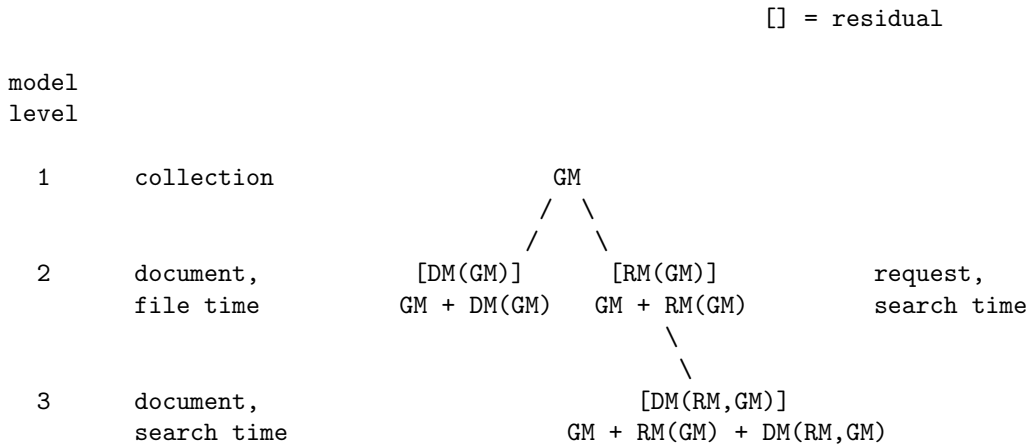

Figure 1: Scope and relations of language models for document retrieval

Assume, then, that we have an approach to language modelling which (a) rewards parsimony, and (b) allows a more specialist model to be built at a lower level than a more general

one. We suppose that every document is generated by a general language or collection model, denoted GM, accompanied by some unknown set of special models relating to topics that occur in the document (and also perhaps in other documents). We will assume that these various topic models can all be rolled into a single individual-document model, denoted DM(GM), which represents the residual: everything in that document that cannot be described by the GM. The complete individual document model thus becomes GM+DM(GM), as illustrated on the left hand side of Figure 1. Further (though not shown in the figure), if we had a model for a specific topic that occurs in the document, this particular topic model would take over part of DM(GM) – potentially more parsimoniously because it is applicable to all documents in which this topic is appears, and because it makes the remaining genuinely-individual document model yet more parsimonious.

We can now express the relevance hypothesis as follows: A request is generated from a specific topic, i.e. relevance, model as usual, to accompany the generic collection model: call this relevance model RM(GM) (so by analogy with the previous case, the full model for relevance is GM+RM(GM)). Then if and only if a document is relevant to the request, the RM(GM) component will apply to this document – that is, it will replace part of DM(GM) in explaining the document. So for a relevant document, we should have a new residual model for all the unexplained parts, DM(GM,RM), and a new complete model GM+RM(GM)+DM(GM,RM), as shown on the right hand side of Figure 1. In this complete model, the RM(GM) component would relate to that part or aspect of the document that makes it relevant to the request.

Thus the probability of relevance of a document is the probability that the complete model just introduced explains the document; more specifically, this is the probability that the GM+RM(GM)+DM(GM,RM) combination is better than the GM+DM(GM) combination. Another way of expressing the relevance hypothesis is as follows: there is a null hypothesis about each document, expressed by the GM+DM(GM) combination; and there is an alternative (relevance) hypothesis, expressed by GM+RM(GM)+DM(GM,RM). We have to test whether the latter explains the document better than the former (where 'better' includes a reward for parsimony); in the usual retrieval fashion, we may express this test by assigning a probability or likelihood to the relevance hypothesis against the null hypothesis.

Now a document that has already been judged relevant provides us with independent evidence, additional to the query, about RM(GM). Thus the multi-level account of the retrieval situation that we have just introduced accommodates the notion of relevance feedback quite naturally. An interesting observation is that the multi-level account actually requires (in principle) that there is more than one relevant document – we can only achieve parsimony if RM(GM) is shared between two or more documents. This is not, however, a practical constraint – any retrieval rule which we could derive from the model would have to be useful in the first stage when we know no relevant documents, and in early feedback when we might have only one; if it subsequently turns out that there is only one or even none in the collection, this will not affect the application of the rule.

The foregoing is clearly only a sketch, but one which suggests some requirements to be met for a comprehensive LM account of retrieval. However, an interesting observation can be made about the approach even at this stage. The process of developing a DM(GM) model for an individual document in the GM+DM(GM) case is independent of any request, and is essentially a file-indexing process: it asks the question "How does the language of this document differ from that of the whole collection?". It looks as if answering this question should deal automatically with stopwords as well as with tf and idf (which one might expect LM to do). Stopwords (or at least some of them) would be words for which the general

language model is sufficient, and which therefore do not figure at all in the residual DM(GM) model for an individual document.

4 Concluding comment

This paper was inspired by the desire to make sense of the relationship between traditional probabilistic models of retrieval and the newer language modelling approach. But this led us to address the role of relevance in the LM approach, which in turn raises issues about the way the LM account of retrieval, when considered from the theoretical rather than the practical point of view, can handle the fact that a request may have any number of relevant documents.

The ideas we have presented are an informal indication of what a suitable LM account of retrieval, i.e. one that recognises the realities, has to be like. It is evident that such an account, implying multiple levels of model, is a good deal more complicated than the attractive basic notion of a (relevant) document generating a request suggests. Our informal analysis clearly needs proper formal development. But it is complicated - indeed disagreeably complex - enough to make doing this quite a difficult enterprise. However it is only when the formal version has been developed that a fair comparison between the LM and PM approaches, as theoretical views of IR, can be properly made. Or, to put the point another way, if both LM and PM are bringing relevance onto the stage in the role of hidden variable, how elegant is her costume, and how convincing are her lines?

References

[1] W. B. Croft and H. R. Turtle. Text retrieval and inference. In P. Jacobs, editor, *Text-based Intelligent Systems*, pages 127–156. Hillsdale, NJ: Lawrence Erlbaum, 1992.

[2] D. Hiemstra. *Term-Specific Smoothing for the Language Modeling Approach to Information Retrieval: The Importance of a Query Term. Proceedings of the 25th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02)*, 2002.

[3] H. Jin, R. Schwartz, S. Sista, and F. Walls. Topic tracking for radio, tv broadcast, and newswire. In *Proceedings, Eurospeech 99*, Vol. VI, pages 2439-2442, 1999.

[4] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization. *Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, pages 111–119, 2001.

[5] V. Lavrenko and W. B. Croft. Relevance-based language models. *Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, pages 120–128, 2001.

[6] D. R. H. Miller, T. Leek, and R. M. Schwartz. A hidden Markov model information retrieval system. *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 214–221, 1999.

[7] S. Mizzaro. Relevance: The whole story. *Journal of the American Society for Information Science*, 48(9), pages 810–832, 1997.

[8] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 275–281, 1998.

[9] J. M. Ponte. Language models for relevance feedback. In W.B. Croft, editor, *Advances in information retrieval : recent research from the Center for Intelligent Information Retrieval*, pages 73–95. Dordrecht: Kluwer, 2000.

[10] T. Saracevic. Relevance: A review of and a framework for the thinking on the notion in information science. *Journal of the American Society for Information Science*, 26, pages 321–343, 1975.

[11] K. Sparck-Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiments. Parts 1 and 2). *Information Processing & Management*, 36(6), pages 779–840, 2000.

[12] M. Spitters and W. Kraaij. A language modeling approach to tracking news events. *Proceedings of the DARPA Topic Detection and Tracking Evaluation Workshop*, 2000.

[13] H. R. Turtle and W. B. Croft. Inference networks for document retrieval. *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, pages 1-24, 1990.

[14] F. Walls, H. Jin, S. Sista and R. Schwartz. Probabilistic models for topic detection and tracking. *Proceedings, ICASSP 99*, 1999.