Part III

Implicit Feedback Frameworks

In Part II I described content-driven techniques that use a variety of document representations to facilitate an increase in the quality and quantity of interaction with retrieval systems. Techniques for selecting query-relevant Top-Ranking Sentences were presented and the results of three studies that tested the effectiveness of the approach were described. The success of the implicit feedback techniques described in Chapter Four encouraged me to enhance these methods. In this part I present two implicit feedback frameworks that use interaction with the interface described in Chapter Five to infer information needs and select retrieval strategies. One of the frameworks is heuristic-based and the other is probabilistic. Part III concludes with a novel simulation-based comparative evaluation of the term selection models in the frameworks and other models. The simulation emulates searcher interaction with content-rich interfaces and serves a formative evaluation technique to establish the most effective implicit feedback model to be tested in later experiments.

Chapter 6

Heuristic-Based Framework

6.1 Introduction

In this chapter the first of two implicit feedback frameworks introduced in this thesis is described. Both frameworks use unobtrusive monitoring of interaction to proactively support searchers. The framework chooses terms to better represent information needs by monitoring searcher interaction with different representations of top-ranked documents. As suggested in Chapter Two, information needs are dynamic and can change as a searcher views information. The framework proposed gathers evidence on potential changes in these needs through changes in term lists used for query formulation by the search system. The framework uses the evidence it gathers to choose new retrieval strategies such as re-searching the document collection or restructuring already retrieved information. Large estimated changes in information need lead to more severe interface support. The framework described in this chapter is heuristic-based and uses term presence/absence in viewed representations to select terms for query modification.

6.2 Information Need Detection

In this section I describe the *Binary Voting Model*, a heuristic-based implicit feedback model I develop to implicitly select terms for query modification. The approach utilises searcher interaction with the document representations and relevance paths described in the previous chapter. The representations viewed by a searcher are used to select new query terms and in the Binary Voting Model each representation 'votes' for the terms it contains. When a term is present in a viewed representation it receives a 'vote', when it is not present it receives no vote. [14] All non-stopword, non-stemmed terms in the top-ranked documents are candidates in the voting process; these votes accumulate across all viewed representations. The assertion I make is that the winning terms are those with the most votes, and hence best describe the information viewed by the searcher. I assume that useful terms will be those contained in many of the representations that the searcher chooses to view. The rationale behind this assertion is that searchers will try to maximise the amount of relevant information they view during a search (Pirolli and Card, 1995). The non-stopword terms that appear in the representations they view (and in similar contexts to their original query terms) are those that are potentially important to the searcher and may be useful for query modification.

6.2.1 Indicativity

In the Binary Voting Model terms are assigned a weight of one or zero, depending on whether they occur in a representation, regardless of the type of representation. All items presented to the searcher at the content-rich search interface are representations of the top-ranked documents. Different types of representation vary in length, and can hence be regarded as being more or less *indicative* of the content of the document. The *length hypothesis* (Marcus *et al.*, 1978) suggests that the quality of a representation is directly proportional to its length. That is, longer representations are regarded as being of a higher quality than shorter representations, simply because they reveal more of the document content. For example, a top-ranking sentence is less indicative (and therefore of a lower quality) than a query-biased document summary (typically composed of four sentences) as it contains less information about the content of the document. The model *weights* the contribution of a representation's vote based on its indicative worth. For example, I consider the contribution that viewing a top ranking sentence makes to the system's understanding of which terms are relevant to be less than a summary.

The weights used in the framework are 0.1 for title, 0.2 for Top-Ranking Sentence (TRS), 0.3 for Summary, 0.2 for Summary Sentence and 0.2 for Sentence in Context. For example all terms in a viewed summary will receive a weight of 0.3; all terms in a viewed summary sentence will receive a weight 0.2, etc. These weights were defined for experimental purposes and were based on the *typical* length of a representation, not their potential semantic

[14] The decision to use binary (term presence/absence in a representation) rather than term frequency (*tf*) information was taken for reasons of simplicity and computational expense. Through empirical investigation I tested the effectiveness of other methods of term weighting such as *tf*, *tf.idf*, *tf* normalised by representation length, none of which performed better than binary voting, and in the case of *tf.idf*, performed worse. This could be because it also included the importance of a term across all document representations relevant or not (i.e., the *idf* weight), not just those viewed.

value. They ensure that the total score for a term is between zero and one (inclusive) and are used in the absence of a more formal methodology.

6.2.2 Term Weighting

The model is a simple approach to a potentially complex problem. The terms with most votes are those that are taken to best describe the information viewed by the searcher (i.e., those terms that are present most often across all viewed representations) and can therefore be used to approximate searcher interests. Of course, searchers may view irrelevant information as they search. In general however, their interaction decisions are guided by a desire to maximise the amount of relevant information they view.

Each document is represented by a vector of length *n*; where *n* is the total number of unique non-stopword, non-stemmed terms in the top-ranked Web documents. [15] In this chapter the list holding these terms is referred to as the *vocabulary*. All terms in the vocabulary are candidates in the voting process.

To weight terms a document × term matrix, shown in Figure 6.1, $(d+1)\times n$ is constructed, where d is the number of documents for which the searcher has travelled at least part of the relevance path. Each row in the matrix represents all n terms in the vocabulary [i.e., $(t_{k1},t_{k2},\ldots,t_{kn})$ where k is the row number], and each term has a weight. An additional row is included for the query.

$$\begin{array}{c|cccc|}
 & t_1 & t_2 & \ldots & t_n \\
\hline
Q_0 & t_{01} & t_{02} & \ldots & t_{0n} \\
D_1 & t_{11} & t_{12} & \ldots & t_{1n} \\
D_2 & t_{21} & t_{22} & \ldots & t_{2n} \\
 & \ldots & \ldots & \ldots & \ldots \\
D_d & t_{d1} & t_{d2} & \ldots & t_{dn} \\
\hline
\end{array}$$

Figure 6.1. Document × Term matrix.

Query terms are initially assigned a weight of one if they are included in the query and zero if not. Example 6.1 (used throughout this chapter) illustrates the operation of the Binary Voting Model.

[15] I do not use word stems since they may not be interpretable by searchers unfamiliar with stemming.

Example 6.1: Simple Updating

If one assumes that there are only 10 terms in the vocabulary and that the original query (Q_0) contains t_5 and t_9, the document × term matrix initially looks like:

$$\begin{array}{c|cccccccccc|} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & t_9 & t_{10} \\ Q_0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}$$

Each row in the matrix is normalised to give each term a value in the range [0, 1] and make the values sum to one. This ensures that the query terms are not weighted too highly in the document × term matrix. This is important when the model is *replacing* query terms; a high query term weight would lessen the chances of other terms being chosen. The matrix now looks like:

$$\begin{array}{c|cccccccccc|} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & t_9 & t_{10} \\ Q_0 & 0 & 0 & 0 & 0 & .5 & 0 & 0 & 0 & .5 & 0 \end{array}$$

Each document representation is regarded as a source of terms, and the act of viewing a representation as an implicit indication of relevance. When a searcher visits the first representation for a document a new row is added to the document × term matrix. This row is a vector of length *n*, where *n* is the size of the vocabulary and all entries are initially set to 0. If a term occurs in a representation, no matter how many times, it is assigned a weight, w_t, which is based on the representation that contains the term.

This weight for each term is *added* to the appropriate term/document entry in the matrix. Weighting terms is therefore a *cumulative* process; the weights calculated for a term in one representation are added to the weights calculated for the preceding steps in the relevance path. Unlike standard RF algorithms which calculate one set of weights for query modification terms between documents, the Binary Voting Model calculates weights on a per document basis (i.e., within documents). There are different sets of weights for each document and these weights correspond to a row in the document × term matrix.

The total score for a term in a document is computed by:

$$w_{t,D} = \sum_{i=1}^{N} (w_{t,r}) \tag{6.1}$$

Where N is the number of steps taken in a relevance path, i is the current step number, D is the document, t is the term, r is the representation and $w_{t,r}$ is the weight of t for representation r.

Example 6.1: Simple Updating (continued)

When a searcher follows a relevance path, the model updates the weights in the document × matrix after each step. Figure 6.2 shows how the term weights are updated as a path from a top-ranking sentence, to title, to summary is traversed.

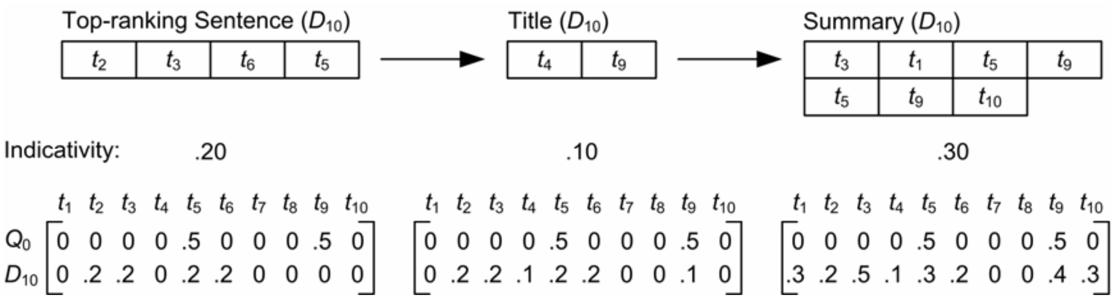

Top-ranking Sentence (D_{10})
t_2 t_3 t_6 t_5
Title (D_{10})
t_4 t_9
Summary (D_{10})
t_3 t_1 t_5 t_9 t_5 t_9 t_{10}
Indicativity: .20 .10 .30
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10}
Q_0 0 0 0 0 .5 0 0 0 .5 0
D_{10} 0 .2 .2 0 .2 .2 0 0 0 0
0 0 0 0 .5 0 0 0 .5 0
0 .2 .2 .1 .2 .2 0 0 .1 0
0 0 0 0 .5 0 0 0 .5 0
.3 .2 .5 .1 .3 .2 0 0 .4 .3

Figure 6.2. Updating the Document × Term matrix.

The weights of all terms except t_7 and t_8 are directly updated. The terms whose weights update are seen as being more important than before to D_{10}. If the document × term matrix is as shown on the far right of Figure 6.2 and the searcher expresses an interest in the title of document D_5 – with a step weight of 0.1, containing terms t_3 and t_5 – the matrix changes to:

$$\begin{array}{c|cccccccccc} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & t_9 & t_{10} \\ \hline Q_0 & 0 & 0 & 0 & 0 & .5 & 0 & 0 & 0 & .5 & 0 \\ D_{10} & .3 & .2 & .5 & .1 & .3 & .2 & 0 & 0 & .4 & .3 \\ D_5 & 0 & 0 & .1 & 0 & .1 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Figure 6.3. Document × Term matrix after addition of new document, D_5.

If the searcher visits one representation of a document and then goes onto the next representation in the path of that document, *at any time – not necessarily immediately*, the model adds the term scores to the row in the matrix occupied by that document. The scoring is cumulative; if a document already has a row in the matrix it does not get a new one.

Similarly, if the searcher views the same representation twice, i.e., the same summary twice, the heuristic-based framework only counts the representation once. The framework in effect, keeps a history of which representations have been viewed; it does not consider more detailed interaction. It is not possible to differentiate, for example, between a searcher seeking relevant information and a searcher checking what they have already examined, something that may account for them looking at the same representation twice.

The matrix resulting from this process reflects the weights based on all paths viewed by the searcher. This information is used for query modification, as will be described in the next section.

6.2.3 Query Modification

In the matrix created by the Binary Voting Model, only the query terms and terms in representations viewed by the searcher will have a score greater than zero. The latter set of terms is potentially useful for query modification.

After every five relevance paths a new query is constructed. This number of paths was established through pilot testing and allows the model to gather sufficient implicit evidence from searcher interaction. It is possible for a relevance path to contain only one representation. Therefore, for the searcher to follow five paths they need only view five unique document representations.

To compute the new query the framework calculates the average score for each term across all documents (i.e., down each column in the document × term matrix). This gives an average score for each term in the vocabulary. The terms are then ranked by their average score. A high average score implies the term has appeared in many viewed representations and/or in those with high indicative weights across the documents viewed. The top six ranked terms are used modify the query. This modification can occur in two ways: *query expansion* and *query replacement*.

Query expansion – The top six terms chosen by the Binary Voting Model are appended to the original terms chosen by the searcher.

Query replacement – It is possible that the new query may not contain the searcher’s original query terms; this would be a form of query replacement as the estimated information need has changed sufficiently to warrant the original query being completely replaced.

Example 6.1 (continued)

If for each term in the ten word vocabulary I average down all rows in the matrix the final weights for each of the ten terms in the vocabulary will now be:

$$\begin{array}{r|cccccccccc|}
 & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & t_9 & t_{10} \\
\hline
Q_0 & 0 & 0 & 0 & 0 & .5 & 0 & 0 & 0 & .5 & 0 \\
D_{10} & .3 & .2 & .5 & .1 & .3 & .2 & 0 & 0 & .4 & .3 \\
D_5 & 0 & 0 & .1 & 0 & .1 & 0 & 0 & 0 & 0 & 0 \\
\hline
\text{average} & .1 & .07 & .2 & .03 & .3 & .07 & 0 & 0 & .3 & .1 \\
\hline
\end{array}$$

The rank order of these terms after computation of these weights is t_5, t_9, t_3, t_{10}, t_1, t_2, t_6, t_4, t_7, t_8. Terms t_5 and t_9 are query terms and since the representations used in this model typically contain query terms I would expect these to be high ranked. In query expansion the top six terms would be added to the original query. In query replacement, the terms t_3, t_{10}, t_1 and t_2 would be appended to t_5 and t_9 to form a new search query.

The terms with the highest scores are those that are present most often in the information viewed by the searcher. The Binary Voting Model assumes that the non-query terms from these can be useful to represent the interests of the searcher. That is, the model considers terms with high weights are important and useful for query modification. The Binary Voting Model generates lists of terms at different temporal locations. The framework described in this chapter uses changes in the ordering of these term lists to estimate potential changes in the topic of the search. In the next section this process is described.

6.3 Information Need Tracking

The framework uses a history of recent interaction to predict changes in the information need of the searcher and make search decisions that may be useful in their search. This history provides insight into the recent interests of the searcher, and by comparing this with previous histories it can be used to track possible changes in the topic of the search. Selecting the most appropriate form of support depends on estimating the extent to which the need changes during a search; the smaller the change, the less radical the support offered. Tailoring the support in this way allows the interface to work in concert with the searcher. The degree of change between successive term lists (formed every five paths) provides evidence to approximate the degree of change in a searcher's information need.

In the set of most-relevant retrieved documents the vocabulary is static, so the framework can gauge the potential level of change in the information need by comparing the change in the term ordering from the term list at step *m* (i.e., L_m) and the list at the subsequent step *m*+1 (i.e., L_{m+1}). The term lists contain all terms in the vocabulary, ranked based on the weights assigned by the Binary Voting Model. As the vocabulary is static, the *terms* in the list will not change, only their order. So, by comparing L_m against L_{m+1} based on some operator $\bigcirc$ the framework can compute the degree of change between the lists and predict possible changes in the information need. This can be shown formally as:

$$\Delta\psi = (L_m) \bigcirc (L_{m+1}) \tag{6.2}$$

Where ψ is the system's view of the searcher's information need and $\bigcirc$ computes the difference between two lists of terms.

The *Spearman rank-order correlation coefficient* is used in this framework as the operator $\bigcirc$. This coefficient tests for the degree of similarity between two lists of rankings. The Spearman rank-order correlation coefficient is non-parametric, so rankings, not the actual term scores, are used. There are two lists of terms, L_m and L_{m+1}, created by the Binary Voting Model at two successive points in time. The first list is ordered by average term score; the second list contains the terms in the same order but updates the rankings (i.e., a new ranking is assigned, but there is *no sorting*). This is shown in Figure 6.4.

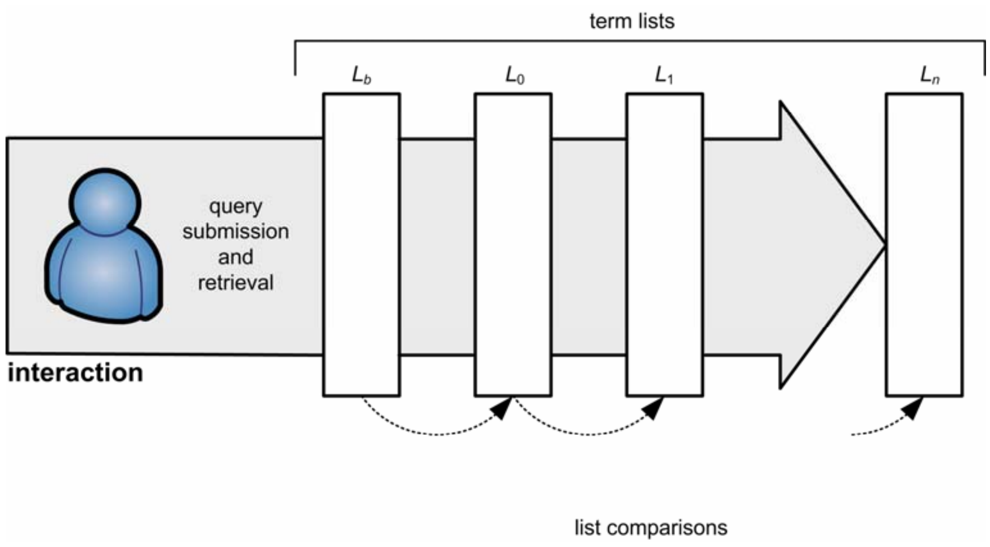


Figure 6.4. Changes in rank order of terms in consecutive term lists.

The Spearman rank-order correlation coefficient returns values between –1 and 1, where 1 is *perfect positive correlation* (the lists are exactly the same), –1 is *perfect negative correlation* (the lists are the complete opposite i.e., 1,2,3,4,5,6,7,8 vs. 8,7,6,5,4,3,2,1) and any value in-between is reflective of their relation to these extreme values. A correlation of 0 implies *zero* (or no) *correlation* between the two lists. Using the Spearman rank-order correlation coefficient $\Delta\psi$ is calculated as follows:

$$\Delta\psi = \frac{\sum_{i=1}^{n} r\left(L_{m_i}\right) r\left(L_{(m+1)_i}\right) - n\left(\frac{(n+1)}{2}\right)^2}{\left(\sum_{i=1}^{n} r\left(L_{m_i}\right)^2 - n\left(\frac{(n+1)}{2}\right)^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{n} r\left(L_{m+1_i}\right)^2 - n\left(\frac{(n+1)}{2}\right)^2\right)^{\frac{1}{2}}} \qquad (6.3)$$

I assume that L_m and L_{m+1} are both ranked lists of terms, $r(.)$ is the rank of a term from one of the lists and n is the total number of terms. Ties are handled in the standard statistical way, by summing the rank of all tied elements and dividing this sum by the number of elements, effectively taking the average rank for each group of ties.

All terms in the original vocabulary [16] are ranked based on the weights derived from the Binary Voting Model, and averaged across all viewed documents. These terms are present in both lists (L_m and L_{m+1}) but potentially in a different order, depending on the representations viewed by the searcher. There is a high level of redundancy in each list as the lower ranking terms that never appear in a viewed representation experience only slight changes in their ranking between iterations. To counter this problem only the top 100 terms are used. These are the most liable to change and hence most likely to reflect any change in the information need. As the number of terms increases (i.e., greater than 100), redundancy in the term list also increases and the predicted level of change becomes more conservative. In contrast, as the number drops (i.e., less than 100) the likelihood of change increases, making the prediction more dramatic.

Term lists are compared every time a new query is created (i.e., every five relevance paths). To compute the correlation coefficient both lists must contain the same terms and the same number of terms. Therefore, in practice the first 100 terms plus β are used. Beta (β) is the number of terms that have left or joined the top 100 terms between L_m and L_{m+1}. For terms *joining* the top 100, these terms are sorted based on their original (L_m) ranks and assign them ranks (in L_m) in the range [101,101 + β]. The same procedure is used for terms that are *leaving* the top 100, except these terms are ranked based on their new (L_{m+1}) ranks (Figure 6.5).

[16] The list of all unique, non-stemmed, non-stopword terms present in the top retrieved documents.

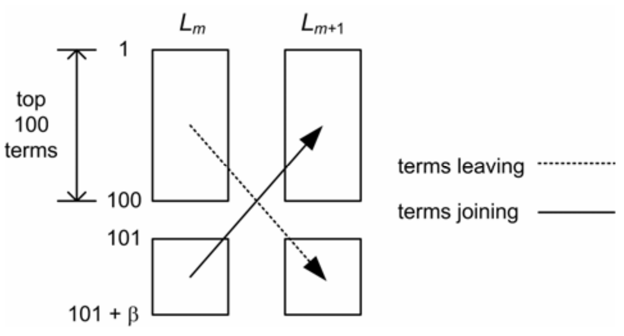


Figure 6.5. Terms leaving and joining the top 100 terms.

The Spearman coefficient is in the range [–1, 1], where a result closer to –1 means the term lists are dissimilar with respect to their rank ordering. Boundaries were chosen for the Spearman coefficient that allowed the need tracking component to choose retrieval strategies. These boundaries are shown in Figure 6.6.

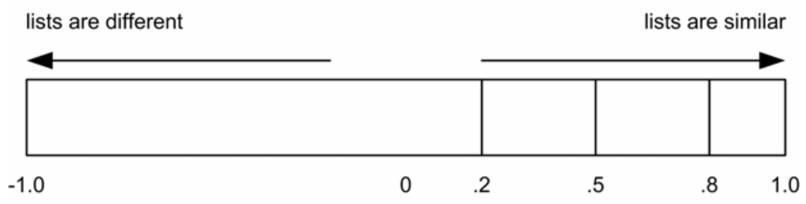


Figure 6.6. Decision boundaries of Spearman coefficient for retrieval strategy selection.

As the coefficient gets closer to one, the similarity between the two query lists increases and based on the coefficient value the framework decides how to use the new list of terms. Four retrieval strategies were implemented:

Re-searching – If the coefficient value indicates that the two term lists are substantially different with respect to rank ordering, I take this to reflect a large change in ψ (the system's formulation of the information need). In this case, a search system implementing the framework will re-search the document collection to retrieve a new set of documents. Coefficient values of less than 0.2 are taken to indicate a large change in the term lists.

Reordering documents – A result in the range [0.2, 0.5) indicates a weak correlation between the two lists and consequently a less substantial change in ψ. Here an implementing search system will use the new query (i.e., the six top ranked terms) to reorder the most-relevant retrieved documents. The document list is reordered using best-match *tf.idf* scoring with the revised query. The vocabulary list remains unchanged after this action.

Reordering Top-Ranking Sentences – Coefficients in the range [0.5, 0.8) indicate a strong correlation between the two term lists and hence a small change in the system's estimation of information needs. In this case the framework uses the new query to re-rank the list of Top-Ranking Sentences. The sentences are the most granular elements presented to the searcher and are therefore most suited to reflect minor changes in ψ. The Top-Ranking Sentences are reordered based on the term-occurrence of each of terms in the new query.

No action – The previous strategies provide an updated view of the retrieved documents based on the current ψ. For differences between 0.8 and 1 (inclusive), the need is assumed to have not changed sufficiently to warrant action.

All numerical bounds are experimental, chosen during pilot testing of the framework. This involved testing an experimental system that implemented the framework interactively with different levels of search topic change (i.e., viewing information on one topic then looking at another). As I viewed information at the results interface the actual value of the Spearman correlation coefficient and boundaries assigned were displayed graphically in a small window in the results interface (Figure 6.7).

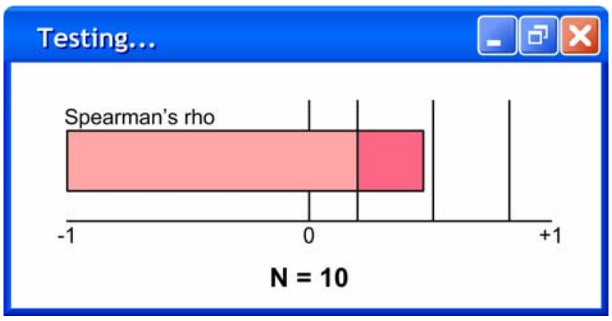


Figure 6.7. Monitoring Spearman rank correlation coefficient during pilot testing.

The lines representing the boundaries could be dragged to different values. Over time, and a variety of search topics, the boundaries were placed in a location that resulted in a high proportion of system decisions being deemed appropriate. This was subjective but was tested in Pilot Test 1 described in Chapter Nine, Section 9.2.1.

6.4 Summary

In this chapter a heuristic-based implicit feedback framework for estimating information needs during a search has been described. The approach uses a Binary Voting Model to create a modified query and a decision metric based on Spearman's rank order correlation coefficient to predict changes in the topic of the search and make new search decisions. The evaluation of this framework using human subjects is described in Chapter Nine (Pilot Test 1). In the next chapter a probabilistic framework for selecting additional terms and retrieval strategies is presented. This framework is potentially more robust than that presented in this chapter and formalises some of the heuristics used.

Chapter 7

Probabilistic Framework

7.1 Introduction

In this chapter a second implicit feedback framework is proposed to estimate current information needs and changes in these information needs during a search session. This framework uses interaction with document representations and relevance paths in the same way as the heuristic-based framework described in Chapter Six. Also, in a similar way to that framework, the techniques presented create modified query statements based on implicit evidence and includes a component to predict when, and by how much, information needs have changed. The approach to select terms is probabilistic and uses *Jeffrey's rule of conditioning* (Jeffrey, 1983) to revise the probability of term relevance in light of evidence gathered from searcher interaction; this is called the *Jeffrey's Conditioning Model*. Jeffrey's conditioning captures the uncertain nature of implicit evidence, and is used since even after the 'passage of experience' the model is still uncertain about the relevance of a term. The approach used for this revision is based on that proposed by Van Rijsbergen (1992). The information need tracking component to estimate need change uses Pearson's correlation coefficient and the statistical significance of this coefficient as a decision metric. I describe the information need detection and need tracking components in Sections 7.2 and 7.3 respectively.

7.2 Information Need Detection

This component uses interaction with representations of top-ranked retrieved documents and relevance paths to predict the interests of searchers. The Binary Voting Model (Chapter Six) used a set of pre-defined heuristic weights for the indicativity of a relevance path's constituent representations. The information need detection component in the probabilistic model replaces this with a measure to describe the value, or worth, of the evidence in a document representation. It combines a confidence measure that uses the relative position of representations in a relevance path with a measure of indicativity based on the concepts in a representation. Unlike the Binary Voting Model, the probabilistic model uses relevance paths directly in the revision of term probabilities. In this section I describe each measure, and how the probabilistic model weights terms for query modification.

7.2.1 Path Weighting

As described earlier in this thesis, Campbell and Van Rijsbergen (1996) produced an extension of the probabilistic model of retrieval that incorporates an 'ageing' component to term weighting. The component incorporates *when* documents containing the terms were assessed relevant. Searchers follow a path through the document space and terms in documents assessed relevant at an early stage in the search receive a lower weight than terms in recently viewed documents.

In content-rich search interfaces searchers traverse relevance paths between document *representations*. Unlike the work of Campbell and Van Rijsbergen, the representations that comprise the path are smaller than documents, the paths are generally short (i.e., no more than six representations) and the most recent document representation is not necessarily the most relevant. The term selection model described in this section assigns an exponentially increasing relevance profile to aged relevance.

The assumption made by this part of the framework is that the further a searcher travels along a relevance path, the more certain it can be about the relevance of the information towards the start of the path. As the viewing of the *next* representation is exploratory and *driven by curiosity as well as information need* the model is cautious, and hence less confident about the value of this evidence. This confidence, *c*, is assigned *from the start of the path* to each representation *i* using:

$$c_i = \frac{1}{2^i}, \text{ where } i \geq 1 \qquad (7.1)$$

However, Equation 7.1 is asymptotic, and therefore the values of c_i do not sum to one. For this to be used in the information need detection component (which is based on probabilistic principles) the sum of all c_i values must be one. The value of c_i is normalised and the confidence *c* for each representation *i* in a path of length *N* is computed using:

$$c_i = \left(\frac{1}{2^i} + \frac{1}{N.2^N} \right), \text{ where } \sum_{i=1}^{N} c_i = 1 \text{ and } i \in \{1, 2, ..., N\} \tag{7.2}$$

Document representations are weighted based on the confidence in their contribution; those near the start of paths are assumed to drive interaction along the path, and are given more weight than those at the end. The representations are weighted based on their position in the relevance path and for the information they contain. A good document representation should be indicative of the source document and in the next section I describe how indicativity is determined.

7.2.2 Indicativity and Quality of Evidence

In the previous section I described the confidence in the relevance of representations based on their position in the relevance path. The profile assumes that subsequent steps in the relevance path are half as relevant; this is perhaps too severe. The quality of evidence in a representation, or its *indicative worth*, can also affect how confident the framework is about the value of its content. In the Binary Voting Model I used heuristics based on the *typical* length of document representations to measure indicativity. However, titles and Top-Ranking Sentences, which may be very indicative of document content, are short and will have low indicativity scores if their typical length is the attribute used to score them.

In this framework I used the non-stopword terms, or *concepts*, in a representation as a measure of indicativity rather than representation length. The weight of a term *t* in document *d* is calculated using its *normalised term frequency* (Harman, 1986), and normalised so that the sum of all weights in a document is one. The larger this value, the more often it occurs in the document, and the more representative of document content that term can be seen to be. To compute the indicativity index *I* for a representation *r* the weights of a term in a document $w_{t,d}$ were summed for all *unique* terms in *r* such that:

$$I_r = \sum_{t \in r} w_{t,d} \tag{7.3}$$

The I_r ranges between zero and one, is never zero, and is one only if the representation contains every unique term in the document. The indicativity measure is only incremented if there is a match between the unique terms in the document and those in the representation. [17] If one assumes that a document *d* contains the set of terms s and representation of that

[17] This measure is similar to a *Hamming distance* (Hamming, 1950), but uses term *weights*, rather than presence/absence.

document *r* contains the terms $\{t_1, t_4, t_{10}\}$. Since terms t_1, t_4 and t_{10} occur in both the representation and the source document, the indicativity index of *r* would be:

$$I_r = w_{t_1,d} + w_{t_4,d} + w_{t_{10},d}, \text{ where } 0 < I_r \leq 1 \tag{7.4}$$

Relevance paths will contain representations of varying quality. The indicativity of a representation can also be seen as measure of the *quality* of the evidence provided by that representation. The indicativity of a representation is multiplied with the confidence associated with that particular step in the relevance path (from Equation 7.2) to compute the *value* of the evidence. Using these measures helps ensure that the worthwhile representations in each relevance path contribute most to the selection of potentially useful query modification terms. In the next section the approach used to select such terms is described.

7.2.3 Term Weighting

The probabilistic model assumes the existence of a *term space T*, a mutually exclusive set of all (non-stemmed, non-stopword) terms in the set of top-ranked retrieved documents. Each term in *T* is independent and has an associated frequency in the top documents. The normalised term frequency of each term in *T* is used to estimate its probability. The probability that a term *t* is relevant based on a probability distribution *P* over *T* as:

$$P(t) = \frac{ntf(t)}{\sum_{t \in T} ntf(t)} \text{ where } ntf(t) = \frac{\log_2(tf(t)+1)}{\log_2 |T|} \tag{7.5}$$

where *ntf* (*t*) is the *normalised term frequency* (Harman, 1986) of term *t* in the term space *T*.

To update this probability based on new evidence gathered from interaction the component uses *Jeffrey's rule of conditioning* (Jeffrey, 1983), applied at the end of each relevance path. Jeffrey's rule is a generalisation of Bayesian belief revision. The basis for Bayes' Theorem (Bayes, 1763) is:

$$P(H \mid e) \propto P(e \mid H)\, P(H) \tag{7.6}$$

or

$$P(H \mid e) = \frac{P(e \mid H) P(H)}{P(e)} \text{ since } \sum_H P(H \mid e) = 1 \tag{7.7}$$

Where *H* is a hypothesis that is supported (or refuted) by some evidence *e* and *P*(*H* | *e*) is interpreted as the probability that this hypothesis is true given *e*. Bayes' Theorem is regarded as a form of belief revision since the component probabilities – $P(e \mid H)$, $P(H)$ and $P(e)$ – are associated with propositions (or events) prior to e being observed and $P(H \mid e)$ is the probability after observation. This can be problematic as Bayesian belief revision requires the evidence e to be certain when it is observed and cannot be disputed (Van Rijsbergen, 1992).

In IR the Bayesian approach is typically used for computing the probability of relevance R given a document and a query through $P(R \mid x)$ where x is some representation of the document assumed to be certain. This means that the description x is assumed to be true of, or in, the document. This may not always be appropriate as the document description may be uncertain at the time of observation. So, for example, a component of a document representation, x_i, might only apply to the source document with a certain probability (Maron and Kuhns, 1960). Since x_i is not certain, Bayes' Theorem cannot compute $P(R \mid x_i)$.

The problem of how to revise a probability measure in light of uncertain evidence or observation was covered by Richard Jeffrey in his book 'The Logic of Decision' (1983). His approach is best described by an example taken from his book.

Imagine a person inspects a piece of cloth by candlelight and thinks that it is green, although it might be blue, or even violet. If G, B and V are the propositions involved then the outcome of the observation might be that the degrees of belief in G, B or V are .70, .25 and .05, whereas *before* observation the degrees of belief were .30, .30 and .40. Represented formally this is:

$$P(G) = .30, \qquad P(B) = .30, \qquad P(V) = .40$$
$$P'(G) = .70, \qquad P'(B) = .25, \qquad P'(V) = .05$$

Here P is a measure of the degree of belief before observation, and P' the measure after observation. The 'passage of experience' has led to P being revised to P', as in $P'(x) = P(x \mid e)$ where e is a proposition, but Jeffrey claims that it is not always possible to express the passage of experience as a proposition. Pearl (1988) used a Bayesian net formalism to make Bayesian conditionalisation appropriate, which was implemented by Turtle and Croft (1992). However, Van Rijsbergen (1992) suggests that Pearl's presupposition of virtual evidence leads to infinite regress (i.e., always being dependent on prior evidence) and that it is better to assume from the beginning that the passage of experience leads to a direct revision of the probability functions.

Given that the person has changed their degree of belief in some propositions G, B, and V as shown above, these changes must be propagated over the rest of the structure of their beliefs. For example, suppose saleability A of the cloth depends on the colour inspection in the following way:

$$P(A \mid G) = .40, \qquad P(A \mid B) = .40, \qquad P(A \mid V) = .80$$

Prior to inspection:

$$P(A) = P(A \mid G)\, P(G) + P(A \mid B)\, P(B) + P(A \mid V)\, P(V)$$
$$= .40 \times .30 + .40 \times .30 + .80 \times .40 = .56$$

After inspection Jeffrey proposes:

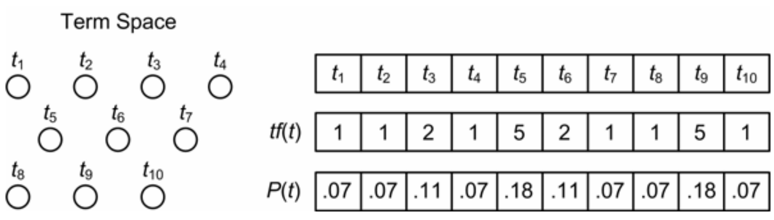
$$P'(A) = P(A \mid G)\, P'(G) + P(A \mid B)\, P'(B) + P(A \mid V)\, P'(V)$$
$$= .40 \times .70 + .40 \times .25 + .80 \times .05 = .0485$$

This is *Jeffrey's rule of conditioning*. This differs from Bayesian conditioning which would use $P'(G) = 1$, or $P'(B) = 1$, or $P'(V) = 1$ and so revise $P(A)$ to $P'(A) = P(A \mid X)$ when $X = G$, B, or V. Bayesian conditioning can therefore be viewed as a special case of Jeffrey's conditioning.

In the context of the work presented in this chapter, a relevance path p is considered as a new source of evidence to update the probability P to P'. I now describe the term weighting approach through an example.

Example 7.1: Simple Updating

Assume the existence of a term space containing 10 terms with the initial values as shown in Figure 7.1. The initial query (Q_0) contains the terms t_5 and t_9. The term space is based on the searcher's query and is therefore created dynamically from the retrieved set of documents; since this set is topically relevant, and the query terms are weighted highly in the initial term space.



	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
$tf(t)$	1	1	2	1	5	2	1	1	5	1
$P(t)$	.07	.07	.11	.07	.18	.11	.07	.07	.18	.07

Figure 7.1. Initial term space frequencies and probabilities for Example 7.1.

The initial $P(t)$ of terms in the term space is determined based on Equation 7.5. This value is normalised to ensure the sum of all probabilities is one. However, it is necessary to revise these probabilities in light of new evidence. In the next section I describe this process.

7.2.4 Probability Revision

The top-ranked documents from which the term space is derived contain a number of document representations. These representations are presented to searchers at the content-rich search interfaces.

The viewing of a representation p_i creates new evidence for the terms in that representation. Let us consider the property of relevance and let us consider the effect of observing an index term t, which is either present ($t = 1$) or absent ($t = 0$). Then:

$$P'(t) = P(p_i \mid t = 1)\, P'(t = 1) + P(p_i \mid t = 0)\, P'(t = 0) \qquad (7.8)$$

This conditional probability may also be estimated through the standard Bayesian inversion using the following formula:

$$P'(t) = \left[P(t = 1 \mid p_i) \frac{P'(t = 1)}{P(t = 1)} + P(t = 0 \mid p_i) \frac{P'(t = 0)}{P(t = 0)} \right] . P(t) \qquad (7.9)$$

This estimation calculates the revised probability of relevance for a term t given a representation p_i, where $P(t = 1)$ is the probability of observing t, and $P(t = 0)$ the probability of not observing t. The prior estimate $P(t = 1)$ is given by collection statistics using Equation 7.5. The probabilities $P'(t = 1)$ and $P(t = 1 \mid p_i)$ are computed in the same way as $P(t = 1)$ (i.e., with Equation 7.5) with one difference in each case; rather than using the frequency of term t in the top documents, $P'(t = 1)$ uses the frequency of t in the whole relevance path and $P(t = 1 \mid p_i)$ uses the frequency of t in the representation p_i. The updated probability $P'(t)$ reflects the 'passage of experience' and is similar to that described by Van Rijsbergen (1992).

A relevance path contains a number of representations. The probabilities are updated after the traversal of a relevance path. The length of a relevance path ranges between one and six steps. I denote this length using *N*. When this length is greater than one the component updates the probabilities across this path. The probability of relevance of a term across a path of length *N* is denoted P_N and given through *successive updating*:

$$P_N(t) = \sum_{i=1}^{N-1} c_i . I_i . \left[\left(P_i(t=1 \mid p_i) \frac{P_{i+1}(t=1)}{P_i(t=1)} + P_i(t=0 \mid p_i) \frac{P_{i+1}(t=0)}{P_i(t=0)} \right) . P_i(t) \right] \qquad (7.11)$$

Where a representation at step *i* in the path *p* is denoted p_i. The confidence in the value of the representation is denoted c_i and I_i is the indicativity of the representation.

In Bayesian belief revision the order of conditioning is irrelevant. However, in Jeffrey's conditioning this is not the case and in general the order the evidence is presented does matter. Therefore the order in which a searcher traverses the relevance path also matters. The content-rich search interface described in Chapter Five restricts the order in which relevance paths can be traversed. However, where a top-ranking sentence appears in the path i.e., in the list of sentences at the start of the path or as a summary sentence, can affect how much that sentence contributes to the selection of terms. A top-ranking sentence contributes more to the selection of new query terms than the same sentence appearing as a 'summary sentence' later in the relevance path; the framework does not weight the same evidence to the same extent twice. This seems reasonable as it was the top-ranking sentence that encouraged the searcher to initiate the traversal of relevance path. The term selection component is uncertain whether further traversal is explorative (i.e., to see what information is available) or verificative (i.e., to verify initial perceptions of relevance) and assigns a reduced weight to representations that fall later in the relevance path to represent this uncertainty.

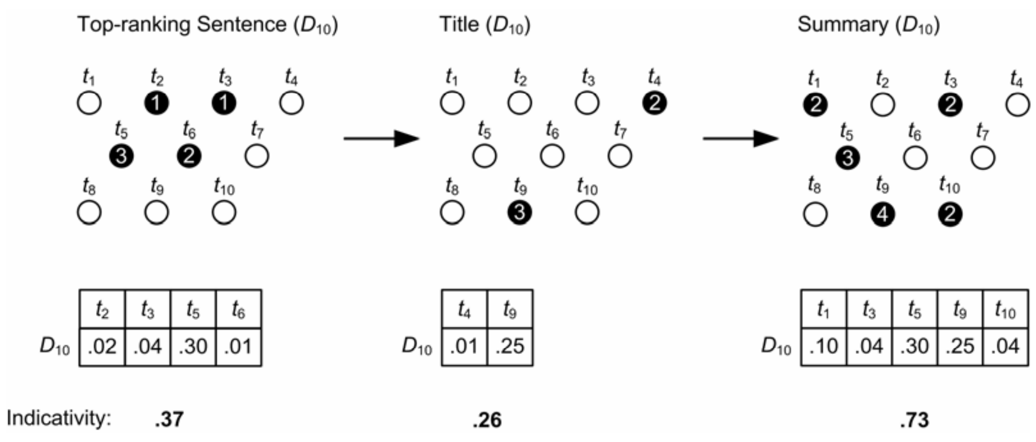
As will be described later in this section, the actual revision of the probabilities will occur after each path. Once learned, the probabilities of relevance remain stable until the next revision (i.e., the next relevance path). Only terms in *T* that appear in the relevance path will have their probabilities revised directly. [18] This order of updating is sequential in nature; that is, relevance paths provide pieces of evidence that are taken sequentially. There is scope for this to allow a searcher to change their mind about the strength of evidence or reverse each revision step. However, this is not implemented in this thesis as revisions occur based on implicit evidence that the searcher may not be aware is being captured or may not be involved

[18] Based on the new evidence probabilities are redistributed to make the sum one.

directly in its provision; they may therefore not know *when* it is appropriate reverse revisions or change the strength of evidence.

Example 7.1: Simple Updating (continued)

When a searcher follows a relevance path, the term selection model updates the weights in the term space after each path. Figure 7.2 shows how the term weights are updated as a path. In this example one can assume the searcher has expressed an interest in a top-ranking sentence from document D_{10}, then the title, and finally the summary. The numbers inside the ball for each term (e.g., ❶) indicate the term frequency in that representation.


Top-ranking Sentence (D_{10})
Title (D_{10})
Summary (D_{10})
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10}
Indicativity:

	t_2	t_3	t_5	t_6
D_{10}	.02	.04	.30	.01

Indicativity: **.37**

	t_4	t_9
D_{10}	.01	.25

.26

	t_1	t_3	t_5	t_9	t_{10}
D_{10}	.10	.04	.30	.25	.04

.73

Figure 7.2. Indicativity at each step in relevance path in Example 7.1.

In this framework the quality of a representation is a measure of its indicativity; how well it represents the source document. A decreasing ostensive relevance profile is applied across the relevance path normalised by the indicativity values shown above in Figure 7.2. This contrasts with the *decreasing* ostensive relevance profile described by Campbell (2000), which assumes that a representation directly following the current representation is half as relevant (shown as a dotted line in Figure 7.3). This seems simplistic, too severe and a normalisation of this profile that includes the quality of the representations in the path is perhaps more fitting. To create such a measure, I multiply the ostensive weight of each step in the relevance path (from Equation 7.2) by its indicativity to form the normalised ostensive relevance (shown by the solid line in Figure 7.3).

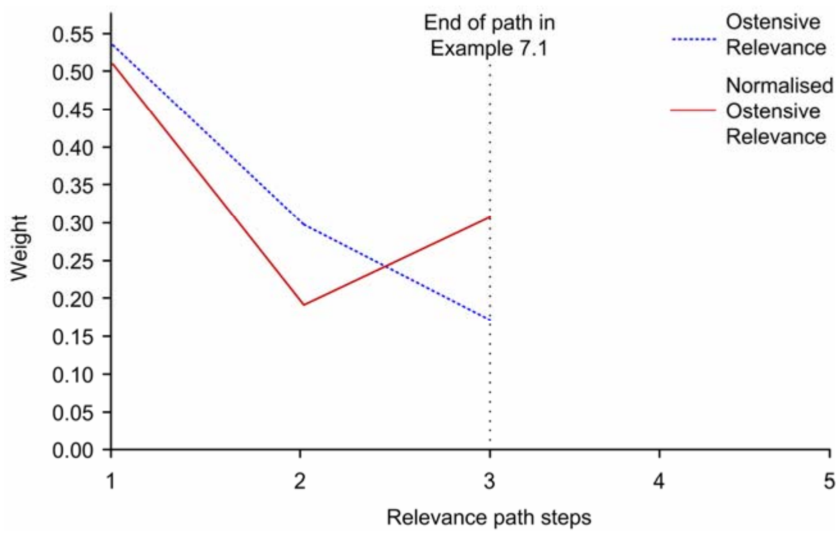
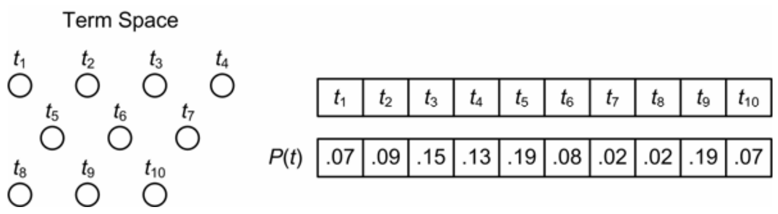


Figure 7.3. Ostensive Relevance and Normalised Ostensive Relevance profiles in Example 7.1.

The weights are stored temporarily in variables until the complete path is traversed. The computation of the revised probabilities is dependent on knowing the length of the relevance path. For this reason it is necessary to wait until the complete path has been traversed before calculating the new set of probabilities. The temporary variables are reset after each relevance path.

The weights of all terms bar t_7 and t_8 are directly updated. Since t_7 and t_8 do not appear in any viewed representations their weights are updated indirectly to ensure the sum of $P(t)$ is one. This can be interpreted as a form of *negative* relevance feedback as the weights of these terms will decrease. Figure 7.4 shows the final state of the term space after all revisions.



	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
$P(t)$	.07	.09	.15	.13	.19	.08	.02	.02	.19	.07

Figure 7.4. Term space in Example 7.1 after relevance path.

The final term ordering, based on the $P(t)$ for each term is t_5, t_9, t_3, t_4, t_2, t_6, t_1, t_{10}, t_7 and t_8 [19]. The terms with the highest scores will be chosen to replace or expand the searcher's original

[19] This is in contrast to the term ordering t_5, t_9, t_3, t_{10}, t_1, t_2, t_6, t_4, t_7, t_8 that arises when the same evidence is presented to the Binary Voting Model in Chapter Six, Example 6.1. The differences in ranking arise because the Jeffrey's Conditioning Model does not simply use term presence or absence and is more sensitive to the *frequency* of terms in document representations.

query. Terms that do not appear in viewed representations have the probability that they are relevant revised downwards. Since t_7 and t_8 do not appear in any of the viewed representations their $P(t)$ is revised indirectly downwards. In contrast, t_3 appears in two of the representations in the relevance path and its $P(t)$ is revised upwards from 0.11 to 0.15.

In this section an approach has been described for estimating the information needs of searchers at a specific point during their search. However, to operate effectively the framework must also be able to identify when a search has changed (i.e., moved from one topic to another). Information needs are dynamic and can change in dramatic or gradual ways as the searcher views information (Bruce, 1994; Robins, 1997). In the next section I describe an information need tracking component similar to that described in Chapter Six that predicts the level of change in a searcher's information need and makes search decisions to help them search effectively.

7.3 Information Need Tracking

In Chapter Six an approach for estimating potential changes in the information needs of searchers was described. This change is based on the differences between the ordered lists of terms, created by the term selection model during the search. Since the order of the terms is affected by the information the searcher expresses an interest in, changes in the term order can be used to track when and by how much a search has changed.

The heuristic approach in Chapter Six uses the Spearman rank order correlation coefficient to provide an estimation of search topic change. The approach worked well, and seemed to choose retrieval strategies that were appropriate. However, in related work (White and Jose, 2004) I have shown that another correlation coefficient, the Pearson product moment (r) also concords with searcher opinion. In this section a similar, but more robust approach is described that uses Pearson's r to estimate search topic change. The statistical significance of r is used to select new retrieval strategies based on the extent of the change and as a further measure of confidence in system decisions.

At every point in the search the contents of the term space are ranked in descending order based on the value of $P(t)$. The order of this term list may change when all $P(t)$ are revised. It is the level of difference between the two lists of terms at different temporal locations that can be used by the framework to predict search topic change.

The information need tracking component in Chapter Six compared consecutive term lists in such a way that the current list of terms was only compared against its immediate predecessor. This meant that the information need would have to change a lot between those iterations for it to be detected by the framework. Rather than watching for changes in information needs over the past few steps, the approach presented in this section looks for changes since the last searcher-defined query iteration. That is, since the searcher last modified their query and used it to generate a new set of search results. The approach uses the interaction that immediately follows the presentation of search results to generate a baseline term ordering (L_b). Each term list (L_i) from that point until the next *re-search* operation is compared to L_b. This is illustrated in Figure 7.5.

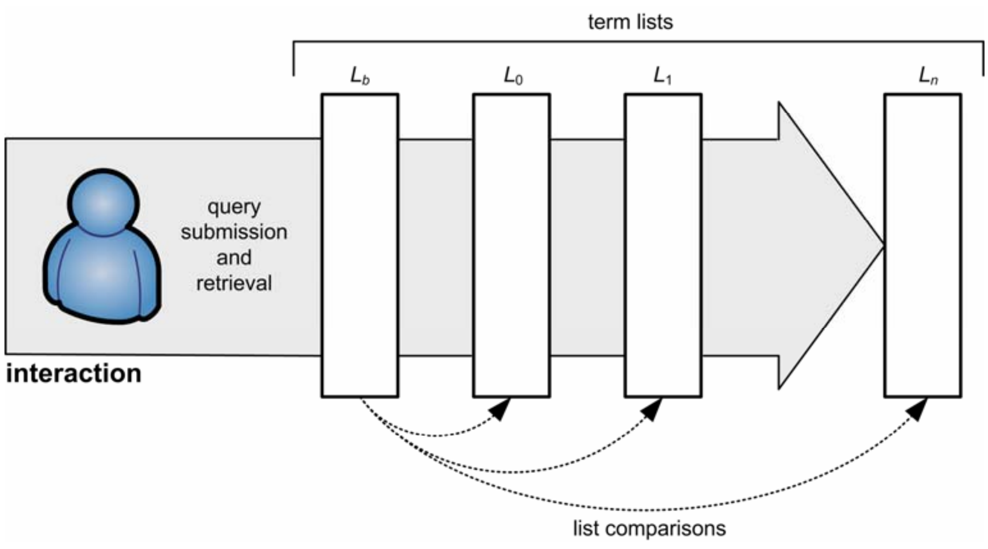


Figure 7.5. Comparing $P(t)$ of terms between term lists.

The approach described in Chapter Six looked for changes in only the top 100 terms in the list. A vocabulary constructed from the top-ranked documents would typically contain some 3000 unique words, depending on the nature of the retrieved documents. This introduces redundancy into the correlation coefficient calculations, as many of the low-ranked terms will experience only slight changes in their ranking during a search session. To address this problem I established a threshold and only used terms whose scores placed them in the top 100. Analysis of the system logs from my experiment found that this threshold was too low, as many terms outside the top 100 changed position also.

The approach proposed in this chapter does not use thresholds; instead it uses the changes in ordering of all terms in viewed representations between query iterations. If a term appears in a viewed representation it is considered an 'active' term and is used in the calculation of

Pearson's r. Terms which are 'inactive', i.e., do not appear in viewed representations, are ignored. I effectively only consider changes those terms whose probability is revised *directly* at some point during the search.

Using correlation coefficients is not the only method that can be used to detect the similarity of the two term lists. Measures of *association* (e.g., Cosine, Jaccard's, Dices), *distance* (e.g., Euclidean, L1(norm)) and *divergence* (e.g., Kullback-Liebler, Jensen-Shannon) can also be used. [20] The disadvantage of these methods is that statistical significance cannot be derived from them. Kupperman (1960) proposed an information statistic based on the χ^2-distribution that estimated the statistical significance of the divergence between two probability distributions using Kullback-Liebler divergence. However, the size and variability of the degrees of freedom (ranging from 0 to 3000) meant that the critical value of χ^2 and hence statistical significance of the information measure, had to be computed in real-time (i.e., it could not be looked up in statistical tables). For reasons of simplicity and reduced computational expense this statistic was not used in this framework. Pilot testing of this statistic for tracking search topic change showed that the statistic could be unreliable, choosing retrieval strategies that were at times inappropriate. In a related pilot study (White and Jose, 2004), the Kullback-Liebler divergence was shown to be a poor predictor of searcher assessments of search topic change. Since the Spearman rank order correlation coefficient (r_s) worked well in Pilot Test 1 (a user experiment described in Chapter Nine, Section 9.2.1) and there was no need to over-complicate this part of the framework. Pearson's r was used in place of r_s as it is parametric, hence based on the values of $P(t)$ not their rank order. In r_s information is lost in the conversion from measurements to rankings meaning that r is more sensitive to small changes in the distribution.

The t-distribution is used to compute the statistical significance of Pearson's r using the formula shown in Equation 7.10 with $N - 2$ degrees of freedom, where N is the number of pairs. The t-distribution is defined as the distribution of the random variable t which is (very loosely) the 'best' that can be calculated not knowing the standard deviation. The test of significance allows us to estimate the probability that there is a relationship between the two term lists.

Pre-defined thresholds are used in a similar way to that shown in Figure 6.6. These techniques are used in conjunction with tests of statistical significance that test the statistical validity of a claim that the difference between the term lists occurred by chance.

[20] For a summary of these measures see Lee (1999).

$$t = \frac{r\sqrt{N-2}}{\sqrt{1-r^2}} \tag{7.10}$$

The null hypothesis states that there is no relationship between the two lists (i.e., $r = 0$). [21] The alternative hypothesis states that there is a true relationship i.e., significantly similar or significantly dissimilar. The value of t and the number of degrees of freedom can be used to predict the statistical significance of the correlation between the two term lists.

The level of significance can be used to help the framework make decisions on which retrieval strategy to employ. The framework must choose one of four possible retrieval strategies (in decreasing order of severity): *re-search Web*, *reorder documents*, *reorder Top-Ranking Sentences* and *no action*. These strategies allow the retrieval system to recreate or restructure the retrieved information based on their most recent estimations of the information need. Firstly the framework obtains a value for r and chooses a strategy in a similar way to that proposed in Chapter Six, although with revised threshold values. [22] It then tests the significance of the correlation. If r is significant at $p < .05$ (two-tailed test) the retrieval strategy proceeds, if not (i.e., $p \geq .05$) then a new retrieval strategy, of lesser severity, is chosen. For example, if the initial decision is to *re-search Web* and:

$N = 20$	$r = .254$
$t(18) = 1.114$	**p = .279**

Since $p > .10$ the decision would be modified to the more conservative option of *reorder documents*. The statistical significance of the difference therefore contributes to the decision about which retrieval strategy should be employed. This happens for all strategies except *no action*, which is not affected since it is the most conservative retrieval strategy. The bounds used by the framework to choose the retrieval strategy are illustrated in Figure 7.6.

[21] If the null hypothesis suggests that r is above or below zero then one must use Fisher's transformation to first make the distribution of r normal, then compute Z-scores and finally p values.

[22] An alternative would be to convert r to r^2 to determine the strength of the correlation. However, this does not consider the direction of the correlation, effectively regarding $-r$ as the same as $+r$. Since a negative correlation can be interpreted as an indication of larger difference the sign of the coefficient is important in this part of the framework.

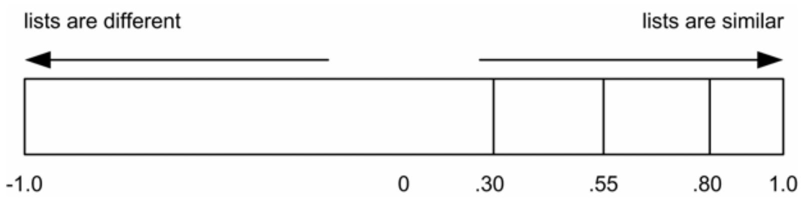


Figure 7.6. The decision boundaries of Pearson's *r*.

The boundaries were *re-search Web* [-1.0, .30), *reorder documents* [.30, .55), *reorder Top-Ranking Sentences* [.55, .80) and *no action* [.80, 1.0]. The boundaries differ slightly from those in the information need tracking component in the heuristic-based framework although the bounds were chosen through pilot testing in a similar way to Chapter Six. The differences between frameworks are attributable to the increased sensitivity of Pearson's *r* (which compares actual values) over Spearman's r_s (which uses rank ordering). For this reason the boundaries originally set to .20 and .50 in Figure 6.6 (Chapter Six) were increased to .30 and .55 respectively, to improve the reliability of the retrieval strategy selection.

To determine the boundaries I used a system that implemented the approach interactively with different levels of search topic change (i.e., looking at information on one topic then looking at another). As I viewed information at the results interface the decision of the system, the actual value of *r* and boundaries assigned were displayed graphically in a small window in the results interface (Figure 7.7).

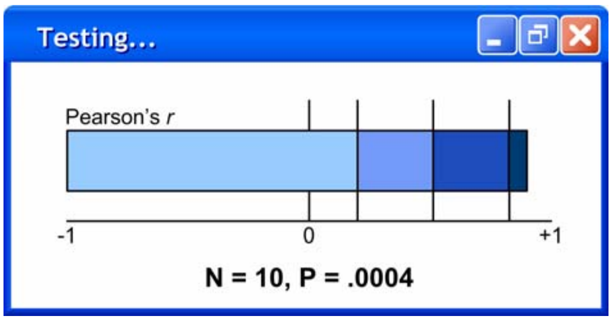


Figure 7.7. Monitoring Pearson's *r* during pilot testing.

The lines representing the boundaries could be dragged to different values. Over time, and a variety of search topics, the boundaries were placed in a location that resulted in a high proportion of appropriate system decisions. These assessments were subjective and the effectiveness of the information need tracking component was further tested in the user experiment described in Part IV.

7.4 Summary

In this chapter a framework for estimating and tracking changes in information needs is presented. The approach uses Jeffrey's rule of conditioning to revise the probability of relevance for all terms in light of new (uncertain) evidence. The most relevant terms are assumed to best represent information needs. In this chapter I have also proposed an approach using Pearson's product moment correlation coefficient to estimate changes in information needs within search sessions and use this estimation to make new search decisions.

In the next chapter I describe a novel simulation-based evaluation that tests the Binary Voting Model described in Chapter Six and the Jeffrey's Conditioning Model described in this chapter. The evaluation compares the performance of these models and other term selection baselines. Later in this thesis I evaluate the implicit feedback frameworks with human subjects; the experimental methodology employed and the results of this experiment are presented in Part IV.

Chapter 8

Benchmarking Implicit Feedback Models

8.1 Introduction

So far in Part III implicit feedback frameworks that use interaction with content-rich interfaces to approximate information needs have been described. In this chapter I introduce a simulation-based evaluation methodology to evaluate the search effectiveness of a variety of term selection (implicit feedback) models independent of searchers. I propose a Simulated IMPLicit Evaluation approach called 'SIMPLE' that simulates interaction with the search interface described in Chapter Five. The models evaluated include, among others, the information need detection components from the heuristic-based and probabilistic frameworks described in Chapters Six and Seven.

There is no standard way to evaluate term selection models that require a lot of searcher interaction with results interfaces. Typically, the only interaction modelled in standard IR experimentation is the provision of relevance feedback through marking relevant documents (Buckley *et al.*, 1994); this is relatively simplistic. The interaction required to provide feedback for the models described in Chapters Six and Seven is complex and a methodology that simulates such interaction is required. Simulation-based methods have been used in previous studies to test query modification techniques (Harman, 1988; Magennis and van Rijsbergen, 1998; Ruthven, 2003) or to detect shifts in the interests of computer users (Lam *et al.*, 1996; Mostafa *et al.*, 2003). These methods are less time consuming and costly than experiments with human subjects, allow environmental and situational variables to be more strictly controlled and complex searcher interactions to be modelled. The SIMPLE approach allows the comparison and fine-tuning of the term selection models before they are employed in an operational IR system. The best performing model is selected and used as the term selection model in the implicit feedback framework tested further in the user experiment described in Part IV. I call components that select additional terms for query modification based on implicit feedback from searchers *implicit feedback models*.

In this chapter I evaluate only the information need *detection* part of the frameworks, not the information need *tracking* component. The information need tracking components are tested through experimentation with human subjects (described in Part IV), not simulations since it may be difficult to assess their effectiveness objectively. Six implicit feedback models are tested using the searcher simulations described in this chapter. These include the Binary Voting and Jeffrey's Conditioning Models from the frameworks described in Chapters Six and Seven and other baseline models. In the next section I describe these baselines.

8.2 Baseline Implicit Feedback Models

In this study I investigate a variety of different methods of relevance feedback weighting based on implicit evidence. The implicit feedback models presented use different methods of handling this implicit evidence and revising their beliefs about searcher needs in light of it. The simulations present the model with evidence in the form of document representations, relevance paths that join representations and the full-text of documents. The study compares the models' ability to 'learn' [23] relevance and create more effective search queries. The performance of the Binary Voting Model (Chapter Six), the Jeffrey's Conditioning Model (Chapter Seven) and four baselines that use a variety of methods to choose additional query terms are compared. Three of the baseline models use the popular *wpq* query expansion method (Robertson, 1990) and one model selects terms randomly. These models are described in subsequent sections.

8.2.1 WPQ-Based Models

The *wpq* method (Robertson, 1990) has been shown to be effective and produce effective query enhancements for query expansion. The equation for *wpq* is shown below, where the typical values r_t = the number of seen relevant documents containing term *t*, n_t = the number of documents containing *t*, *R* = the number of seen relevant documents for query *q*, *N* = the number of documents in the collection.

$$wpq_t = \log \frac{r_t / (R - r_t)}{(n_t - r_t)/(N - n_t - R + r_t)} \cdot \left(\frac{r_t}{R} - \frac{n_t - r_t}{N - R} \right) \tag{8.1}$$

[23] The word 'learn' is used to refer to the process in which the term selection models improve the quality of their query formulations incrementally during a search session creating a ranking in the list of terms in the vocabulary that approximates the term distribution across the set of relevant top-ranked documents.

In the models described in this chapter, whole documents *and* document representations such as titles, summaries and Top-Ranking Sentences, can be considered relevant. The *wpq* method is based on probabilistic distributions of a term in relevant and non-relevant documents. As the values of r_t and R change during searcher interaction, the *wpq*-generated term weights also change. However, there is no retained memory of these term weights between iterations, and wpq_t is recomputed after each iteration. The *wpq* approaches learn what search results are relevant but do not directly 'remember' the weights assigned to *terms*. For example, a model based on *wpq* may be aware which documents have been explicitly marked but since these documents may change the term weights will have to be re-computed from zero at every iteration. In contrast, the Jeffrey's Conditioning and Binary Voting Models, store and revise term weights for the entire search session. At any point the term space, or vocabulary, stores the term weights for all potential query modification terms.

8.2.1.1 WPQ Document Model

The *wpq document model* uses the full-text of documents, rather than granular representations or paths that link them. The *wpq* formula is applied to each document and expansion terms chosen from it. In Equation 8.1 the values of R = the number of seen documents, r_t = the number of seen documents containing term t, N = the number of top-ranked documents and n_t = the number of top-ranked documents containing the term t. This approach is effectively a traditional explicit relevance feedback model, choosing one relevant document per iteration. This is a realistic model since implicit feedback is typically gathered sequentially (i.e., one relevance indication after another) and was included in the study to investigate the effects of using whole documents for such feedback.

8.2.1.2 WPQ Path Model

In the *wpq path model* the terms from each complete relevance path are pooled together and ranked based on their *wpq* score. The values R = the number of seen paths, r_t = the number of seen paths containing term t, N = the total number of paths generated from the top 30 retrieved documents, n_t = the number of generated paths that contain the term t are used in for the variable values in Equation 8.1. Since it uses terms in the *complete path* for query expansion, this model does not use any path weighting or indicativity measures. This model was chosen to investigate combining *wpq* and complete relevance paths for implicit feedback.

8.2.1.3 WPQ Ostensive Profile Model

The *wpq ostensive* [24] *profile model* considers each representation in the relevance path separately, applying the *wpq* formula and ranking the terms each representation contains. This model adds a temporal dimension to relevance, assigning a within-path *ostensive relevance profile* (Campbell and Van Rijsbergen, 1996) that suggests a recently viewed step in the relevance path is more indicative of the current information need than a previously viewed one. This differs from the Jeffrey's Conditioning Model, which assigns a reduced weight to most recently viewed step in the path. The *wpq* weights are normalised using such a profile. The model treats a relevance path as a series of representations, and uses each representation separately for *wpq*. In this model the *wpq* formula uses the values R = the number of seen representations, r_t = the number of seen representations containing term t, N = the number of representations in top-ranked documents, n_t = the number of representations containing the term t. This model uses an ostensive relevance profile to enhance the *wpq path model* presented in the previous section.

8.2.2 Random Term Selection Model

The random term selection model assigns a random score between zero and one to terms from viewed representations. At the end of each relevance path, the model ranks the terms based on these random scores and uses the top-scoring terms to expand the original query. This model does not use any path weighting or indicativity measures. This model is a baseline and was included to test the degree to which using any reasonable term-weighting approach affected the success of the implicit feedback. Also, since it did not retain any memory of important terms or search results this model was also expected to experience no learning.

In the next section I describe the simulated-based evaluation methodology used to test each of the six implicit feedback models.

8.3 Simulation-Based Evaluation Methodology

There has been no precedent set on how to evaluate implicit feedback models. In this study a simulation-based evaluation methodology is used to simulate interaction with the style of results interface described in Chapter Five, to benchmark such models and choose the best performing model to be further tested in the user experiment described in Part IV. This simulation-based study is therefore a formative evaluation of the implicit feedback models, in which only the best model is chosen for further experimentation.

[24] The only similarity to the Ostensive Model of Relevance (Campbell, 2000) is the exponentially increasing relevance weight applied to document representations at subsequent temporal positions.

The simulation assumes the role of a searcher, browsing the results of an initial retrieval. The information content of the top-ranked documents in the first retrieved document set constitutes the information space that the searcher must explore. All interaction in this simulation was with this set and a new set of results was never generated since I want to evaluate the performance of the model between searcher-defined query iterations. In the simulation searchers were modelled using a number of different strategies: (i) assume they only view relevant/non-relevant information, i.e., follow relevance paths from only relevant or only non-relevant documents, (ii) assume they view all relevant or all non-relevant information, i.e., follow all relevance paths from top-ranked relevant documents or top-ranked non-ranked documents, (iii) exhibit differing degrees of ‘wandering’ behaviour, i.e., try to view relevant information but also viewing different amounts of non-relevant information.

The models are tested based on how well they improve search *precision* (the proportion of retrieved documents that are relevant) and ‘learn’ the distribution of terms across the relevant documents. Since searchers typically exhibit a limited interaction with the results of their retrieval (Jansen *et al.*, 2000) it is important to ensure that most of the information they interact with is relevant. For this reason, precision is used as a measure of search effectiveness in this study rather than *recall* (the proportion of relevant documents retrieved).

In this section the evaluation methodology is introduced. The system, corpus and topics used are described in Section 8.3.1. In Section 8.3.2 the techniques used to extract the relevance paths are described and in Section 8.3.3 the different searcher modelling strategies that use the relevance paths are described. In Section 8.3.4 the relevant distributions and correlation coefficients used to evaluate how well the models learn relevance are presented. The procedure and a description of the study are given in Section 8.3.5 and 8.3.6 respectively.

8.3.1 System, Corpus and Topics

The popular SMART search system (Salton, 1971) was used in the experiment to index and search the corpus. The test collection used was the San Jose Mercury News (SJMN 1991) document collection taken from the TREC initiative (Voorhees and Harman, 2000). This collection comprises 90,257 documents, with a mean average 410.7 words per document (including document title), a mean average 55.6 relevant documents per topic and has been used successfully in previous experiments of this nature (Ruthven, 2003). The creation of relevance paths requires documents that contain at least four sentences. However, to create worthwhile paths with well-formed ‘sentences in context’ (see Chapter Five, Section 5.2.5) the component requires documents that contain around ten sentences. [25]

TREC topics 101-150 were used and the query was taken from the short *title* field of the TREC topic description. For each query the top 30 documents are used to generate relevance paths for use in the simulation. The number and nature of relevance paths chosen for the simulation is dependent on the simulation strategy employed, i.e., how the simulated searchers interact and how relevance paths are selected. The simulation assumes that searchers look at a subset of relevant paths, all relevant paths or a mixture of relevant and non-relevant paths. Non-relevant paths are assumed taken from non-relevant documents.

The simulation retrieves the top 30 results for each of the 50 TREC topics used as queries in this study; these results can contain both relevant and non-relevant documents. In some scenarios the simulation requires paths from only non-relevant documents, only relevant documents or a mixture of both. However, for some topics, there are no relevant documents in the top 30 results, making the execution of the latter two scenarios problematic. Therefore, when the simulation uses paths from relevant documents, it uses only those queries that have relevant top-ranked documents (i.e., 43 of the 50 topics have relevant documents in the top 30). There are non-relevant documents in the top 30 for all topics, so the same problem does not arise. I now explain how paths are extracted from top-ranked results for each topic.

8.3.2 Relevance Paths

In the simulation paths are extracted only from relevant documents, only from non-relevant documents or from a mixture of relevant and non-relevant documents, depending on the simulation strategy. Each document has a set number of representations and number of possible relevance path routes between these representations. In Table 8.1 all routes for all path types are shown. The final ‘document’ step is not included in the simulation since it is not used by the implicit feedback models at the search interface.

[25] Documents with only four sentences may result in low quality summaries and sentences in context comprised of other summary sentences, not new sentences that may contain useful alternate terms.

Table 8.1
Possible relevance path routes.

Document Representations					
TRS	Title	Summary	Summary Sentence	Sentence in Context	Total
4	1	1	4	1	16
4	1	1	4		16
4	1	1			4
4	1				4
4					4
	1	1	4	1	4
	1	1	4		4
	1	1			1
	1				1

For example, for viewing all five representations (first row of Table 8.1) there are 4 × 1 × 1 × 4 × 1 = 16 possible paths. The final column shows the total for each possible route. There are 54 possible relevance paths for each document. If all top 30 documents are used there are 1,620 (54 × 30) possible relevance paths per search topic. In the next section more details are given on how search scenarios that use these paths are deployed in the simulation.

8.3.3 Simulated Search Scenarios

To operate effectively the implicit feedback models should handle different retrieval situations. Since the models rely on the interaction of searchers it is necessary to test them with different styles of interaction or retrieval scenarios. To do this, the way in which relevance paths are chosen is varied and the models are tested in *extreme* and *pre-modelled* situations. In this section styles of interaction that represent each of these situation categories are described in more detail. Paths and documents are considered synonymous unless otherwise stated.

8.3.3.1 Extreme Situations

Styles of interaction in this category represent extreme situations where *only* relevant or non-relevant paths are traversed. Two strategies are presented, one where all paths are traversed and another where a subset of these paths is traversed. These strategies create bounds on the performance of the system and model the situation where searchers (by chance) interact *only* with relevant or non-relevant information. They determine the best or worst expected performance of the models, depending on the paths or documents chosen.

8.3.3.1.1 All Paths

This strategy creates relevance paths from all documents in the top 30 retrieved by the search system. Each relevance path is treated in isolation and the effect of paths traversed in sequence is not cumulative. Although queries submitted for different TREC topics retrieve different numbers of relevant and non-relevant top-ranked documents this approach allowed the best and worst performing paths (and sets of paths) for each topic, and across all topics, to be identified. This can be useful to establish the attributes of good and bad relevance paths.

8.3.3.1.2 Subset of Paths

Searchers would typically not view all retrieved information. This strategy randomly selects a subset of paths used in the 'All Paths' situation. Paths are traversed in sequence and the effect across paths is cumulative. That is, unlike the 'All Paths' situation, the term scores in the term selection models are not reset after each path.

8.3.3.2 Pre-modelled Situations

The implicit feedback frameworks described in Chapters Six and Seven assume searchers will try to interact with relevant information, but accept they will inevitably also view information that is non-relevant. Pre-modelled situations model circumstances where searchers may view relevant and non-relevant paths as they explore the retrieved information. This level of 'wandering' is measured as a percentage of the viewed paths that are not from relevant documents. For the purposes of this study these paths were regarded as irrelevant. The effectiveness of the term selection models at different levels of wandering can be tested. The amount of wandering can vary due to search experience or familiarity with the task and the topic of the search. The empirical findings of the user experiment presented in Part IV of this thesis suggests that non-relevant relevance paths contained fewer steps than relevant paths. As will be shown later in this thesis, the Checkbox system in that experiment allowed subjects to assess the relevance of each document representation in a relevance path and explicitly communicate their decisions to the retrieval system. Paths with no relevance assessments were shorter than those with at least one assessment, suggesting that irrelevant paths should be shorter in the simulation. Inferences made from interaction logs can assist in the development of richer simulated search strategies that can better approximate the interaction of real searchers. In Section 8.3.3.3 I use these data to model the length of relevance paths.

It is possible to vary how relevant (R) and non-relevant (N) paths are distributed to test how the models perform in different circumstances. The distribution method described in this section use previously traversed paths to select future paths.

8.3.3.2.1 Related Paths

This method selects paths that are related to those previously followed. The first path to be visited is chosen at random from the list of available paths. This path can be relevant or non-relevant. Subsequent paths are randomised in such a way that for ten paths and 50% wandering the order of traversal may be {*R*, *N*, *R*, *N*, *N*, *R*, *R*, *N*, *R*, *N*}. The paths are traversed from the first path onwards. The method decides whether the path will be relevant or irrelevant using the order of traversal and selects the actual path based on candidate path quality and its similarity to the current path. The *quality* of a relevance path is measured by its indicativity index introduced in Chapter Seven. The index is a measure of how well a document representation represents the concepts in its source document. The degree to which subsequent paths are *related* is computed using the *Pearson product moment correlation coefficient*. This coefficient has been shown to be an effective measure of similarity in a related study with human subjects (White and Jose, 2004). The product of these two measures is used as a decision metric to rank candidate relevance paths and select future paths. The highest ranked candidate path is chosen as the next path to be traversed. The use of this combined measure simulates searchers' desire to view high-quality, related information. That is, the path with the highest aggregate quality and similarity to the current path is the most likely to be traversed next by a simulated searcher. During a search session searchers would typically follow a series of *related* relevance paths in a rational way, viewing only the most useful or interesting. This strategy attempts to simulate this activity.

In {*R*, *N*, *R*, *N*, *N*, *R*, *R*, *N*, *R*, *N*} the path at position two is non-relevant. To select the actual path all candidate non-relevant paths are ranked based on the product of their quality and similarity to the path at position one. The highest ranked path is chosen as the next step and the process repeats until ten paths have been visited in the order described. Pre-modelled situations are potentially more realistic than extreme situations since they make real-time predictions on what paths to follow and do not assume that searchers only interact with relevant information.

8.3.3.3 Path Length Distribution

The modelled situations use empirical evidence to decide that relevance paths taken from irrelevant documents were short, i.e., three steps or less. However, it is possible to further analyse these results and derive another strategy that creates a distribution of path lengths across relevant and non-relevant paths. Data gathered from interactive experimentation with the Checkbox system in Part IV of this thesis allowed the construction of path length distributions. This system allowed subjects to explicitly mark document representations as relevant. In that experiment, relevance paths considered as relevant if one or more of its constituent representations were marked as relevant by experimental subjects. Table 8.2 shows how path lengths are distributed across relevant (containing marked representations) and non-relevant (containing no marked representations) relevance paths.

Table 8.2
Path length distribution in relevant and non-relevant paths (values are percentages).

Steps	Path type	
	Relevant	Non-relevant
1	14.18	23.45
2	9.53	25.76
3	18.95	30.28
4	25.11	13.67
5	32.23	6.84

From these results it appears that searchers interacted differently with relevant and irrelevant information. More specifically, it demonstrates that the paths were longer if they contained relevant information. The values in Table 8.2 can be used in pre-modelled situations to control the number of paths of each length used in the simulation. For example, if there are ten relevant paths and 0% wandering i.e., $\{R, R, R, R, R, R, R, R, R, R\}$, then their would be one path of length one (14.18% of 10), one path of length two (9.53% of 10), two of length three (18.95% of 10), three of length four (25.11% of 10) and three of length five (32.23% of 10). The number of paths of each length is rounded to the nearest integer. These path length distributions may be used to simulate the general behaviour of real searchers when using content-rich interfaces. This can be a robust alternative to choosing paths regardless of length or imposing upper bounds on the length of paths from irrelevant documents.

In all strategies, model performance is measured based on how the modified queries they generate influence search precision. As well as being able to improve search effectiveness (through creating well-formed queries) the models should learn relevance when shown examples of what is relevant. In the next section I describe the use of relevant distributions and correlation coefficients to measure such learning.

8.3.4 Relevant Distributions and Correlation Coefficients

A good implicit feedback model should, given evidence from relevant documents, learn the distribution across the relevant document set. The model should train itself, and become attuned to searcher needs in the fewest possible iterations.

A relevant term space for each topic is created before any experiments are run. This space contains terms from all the relevant documents for that topic, ordered based on their probability of relevance for that topic, computed in the same way as Equation 7.5. After each iteration the extent to which the term lists generated by the implicit feedback model correlates with the relevant distribution is measured. The simulation 'views' relevance paths from relevant documents and provides the models with the implicit relevance information they need to train themselves. I measure how well the models learn relevance based on how closely the term ordering they provide matches the term ordering in the relevant distribution.

To measure this I use two nonparametric correlation coefficients, *Spearman's rho* and *Kendall's tau-b*. These have equivalent underlying assumptions and statistical power, and both return a coefficient in the range [-1, 1]. However, they have different interpretations; the Spearman accounts for the proportion of variability between *ranks* in the two lists, the Kendall represents the difference between the probability that the lists are in the same order versus the probability that the lists are in different orders. I use both correlation coefficients to verify learning trends.

8.3.5 Evaluation Procedure

The simulation creates a set of relevance paths for all relevant and non-relevant documents in the top-ranked documents retrieved for each topic. The use of these paths, how feedback iterations are generated and the number of feedback iterations (*m*) depends on the simulation strategy employed. After each iteration, I monitor the effect on search effectiveness and how closely the terms chosen by the model correlate with the term distribution across that topic's relevant documents. The correlation is a measure of how well the model learns the relevant term distribution and precision is a measure of search effectiveness.

The following procedure is used *for each topic with each model*:

i. use SMART to retrieve document set in response to query (i.e., topic title) using an *idf* weighting scheme and record the initial precision values.

ii. identify relevant or non-relevant documents in the top 30 retrieved documents, depending on the experimental run and store in set *s*.

iii. select Top-Ranking Sentences from all documents in *s* using the approach presented in Chapter Three.

iv. create and store all potential relevance paths for each document in *s* (up to a maximum of 54 per document).

v. choose relevance paths or documents as suggested by the simulation strategy, setting *m* to the number chosen. The Java [26] random number generator is used where appropriate in selecting random paths or documents.

vi. for *each* of the *m* relevance paths/documents:

   a. weight terms in path/document with chosen model and rank terms based on weights.
   b. monitor correlation between terms and topic's relevant distribution.
   c. choose top-ranked terms and use them to expand original query.
   d. use new query to retrieve new set of documents.
   e. compute new precision values.

To better represent a searcher exploring the information space, all simulated interaction was with the results of the first retrieval only. All subsequent retrievals were to test the effectiveness of the new queries and were not used to generate relevance paths. In the next section the simulated study is described.

8.3.6 Simulated Study

A study of how well each term selection model learned relevance and generated queries that enhanced search effectiveness is now presented. The models are tested in extreme and pre-modelled situations and each requires a different evaluation approach. The strategies used either the 43 'useable' topics (only paths from relevant documents or a mixture of relevant and non-relevant documents) or all 50 topics (only paths from non-relevant documents) and added six terms to the original query. This was done without any prior knowledge of the effectiveness of adding this number of terms to queries for this collection. Harman (1988) showed that six terms was a reasonable number of additional terms for use in simulated experiments. Query expansion was used to test the marginal effectiveness of the model i.e., how much each new query improved the retrieval over the query before any modification. A *run* in the study involves the testing of a model under a particular experimental condition. An *iteration* is a single relevance path or document.

8.3.6.1 Extreme Situations

The evaluation strategy used in extreme situations models the situation where searchers have (by chance) interacted with relevant or irrelevant information.

[26] http://java.sun.com

8.3.6.1.1 All Paths

This strategy uses all paths from the top 30 relevant documents and all paths from the top 30 non-relevant documents. A run of the simulation comprised 54*n* relevance paths, where *n* is the number of relevant/non-relevant documents. The correlation coefficients and search effectiveness were measured after each iteration. The effect of term scoring across consecutive paths is not cumulative. That is, paths were treated in isolation. The evaluation investigated performance differences of paths generated (e.g., best path/worst path).

8.3.6.1.2 Subset of Paths

This strategy used a subset of the paths generated in the 'All Paths' situation. I ran the simulation ten times and *each run comprised 20 iterations*. I recorded correlation coefficients and measures of search effectiveness at iterations 1, 2, 5, 10 and 20. This allowed me to monitor model performance at different points in the search. In the document-centric approach each *document* is regarded as an iteration. Therefore, when this approach was used, it was only possible to have as many iterations as there are relevant/non-relevant top-ranked documents.

8.3.6.2 Pre-modelled Situations

Three pre-modelled methods were tested in this study. Unlike the extreme situations these methods do not assume that searchers could only interact with relevant information. The 'Related Paths' method made decisions on what paths to visit based on those traversed previously. In a similar way to the 'Subset of Paths' strategy I ran the simulation ten times for each implicit feedback model and recorded correlation coefficients and measures of search effectiveness at iterations 1, 2, 5, 10 and 20. The level of wandering was varied in each of the models and recorded at 10%, 20%, 30%, 40% and 50%. In the document-centric approach, the minimum amount of wandering was one document. Across all pre-modelled situations the effect of path length could be ignored or path length distributions based on the results of empirical studies used to make more informed path choices.

8.3.6.3 Experimental Scenarios

In this section I describe the eight simulated scenarios that test the implicit feedback models in different situations. Table 8.3 shows these scenarios and the variables changed in each scenario. If a variable varies as part of a scenario a dot (•) is shown in the corresponding cell.

Table 8.3
Experimental scenarios and variation in experimental variables.

Scenario		Paths/Documents		Relevance			Path length distribution	Wandering
Number	Name	All	Subset	*R*	*N*	*R* and *N*		
1	All Paths	•		•				
2	All Paths	•			•			
3a	Subset of Paths		•	•				
3b	Subset of Paths		•	•			•	
4a	Subset of Paths		•		•			
4b	Subset of Paths		•		•		•	
5a	Related Paths		•			•		•
5b	Related Paths		•			•	•	•

Scenarios 3, 4 and 5 are each divided into scenarios ‘a’ and ‘b’. In ‘a’ paths are selected randomly whereas in ‘b’ a path length distribution is used to select paths. In each scenario all six implicit feedback models introduced earlier in this chapter are used to generate new queries. The resultant precision values and correlation coefficients are used to assess the performance of the models. In the next section I describe the results of the simulated study for each experimental scenario with each implicit feedback model.

8.4 Results

The study was conducted to evaluate a variety of implicit feedback models using searcher simulations. In this section I present results of the study for each simulation strategy. In particular I focus on results concerning search effectiveness and relevance learning. I use the terms *bvm*, *jeff*, *wpq.doc*, *wpq.path*, *wpq.ost* and *ran* to refer the Binary Voting, Jeffrey’s Conditioning, wpq document, wpq path, wpq ostensive and random models respectively. All uses of the term ‘average’ in the remainder of this chapter refer to the *mean average*.

8.4.1 Scenario 1: All Relevant Paths

The aim of this scenario was to predict the best and worst performing paths for each model. In this scenario, all extracted paths across all relevant documents for each topic were used on a per-topic basis. For each topic there were 54*n* paths, where *n* is the total number of relevant documents in the top-30 retrieved. In total, there were 15,174 paths (i.e., 54 × 281 [27]) across the 43 topics used in this study. After each path the effect of that path on correlation coefficients was recorded and for each model the 15,174 paths were ranked based on their marginal effect on the Spearman and Kendall correlation coefficients. That is, the paths were ranked independent of source document, based on their ability to increase the rate in which

[27] In total, there were 281 relevant documents in the top 30 retrieved for all 43 search topics used.

the term selection model learned relevance. This allowed me to predict the ten best and worst performing paths and analyse why some paths were good and some were bad. In Tables 8.4 and 8.5 I show the average best and worst path performance for each of the six term selection models. Also included are the marginal effect on correlation (averaged across both coefficients) of each path, the average path length and the indicativity score in relation to the source document and the relevant distribution the model is trying to learn. In these tables I also show total number of terms in a path and in brackets the percentage of those terms that are stopwords (i.e., common words such as ‘a’, ‘the’ and ‘of’).

Table 8.4
Average best path performance in Scenario 1.

Term selection model	Rank order	Marginal Correlation	Length	Number of Terms	Indicativity	
					Document	Distribution
bvm	4	0.580	3.9	186 (45.6%)	0.391	0.076
jeff	1	0.659	3.1	139 (47.0%)	0.448	0.062
wpq.doc	3	0.616	–	–	1.000	0.049
wpq.path	2	0.640	3.9	146 (46.9%)	0.632	0.045
wpq.ost	5	0.529	3.9	158 (45.3%)	0.517	0.049
ran	6	0.503	4.0	172 (47.7%)	0.364	0.062

Table 8.5
Average worst path performance in Scenario 1.

Term selection model	Rank order	Marginal Correlation	Length	Number of Terms	Indicativity	
					Document	Distribution
bvm	4	– 0.278	3.5	141 (48.8%)	0.295	0.045
jeff	1	– 0.219	3.5	168 (44.7%)	0.366	0.043
wpq.doc	6	– 0.594	–	–	1.000	0.033
wpq.path	5	– 0.289	4.3	179 (47.7%)	0.386	0.030
wpq.ost	2	– 0.253	3.1	130 (45.9%)	0.411	0.053
ran	3	– 0.264	4.3	172 (46.7%)	0.323	0.040

The same paths perform differently for different term selection models and only very rarely does the same path appear as the best path for a number of models. The ability of a term selection model to learn what information is relevant is dependent on the paths used. A good term selection model should maximise the rate of learning when shown relevant information, but minimise the negative effects when shown irrelevant information.

From these tests path length, the number of terms and percentage of those terms that were stop words have little influence over path performance. However the indicativity, or *quality*, appears different between good and bad performing paths. I can conjecture from this that paths that lead to poor term selection model performance are not indicative of their source documents or the relevant term distribution for the TREC topic they were created relative to. These results also describe the best and worst possible correlation values for each of these models. The Jeffrey's Conditioning and wpq.path models performs best, as they have the highest potential marginal gains in correlation coefficients and the lowest potential marginal losses for selecting random path from the set of all paths.

8.4.2 Scenario 2: All Non-Relevant Paths

This scenario was very similar to Scenario 1 but used paths from non-relevant documents rather than relevant. This was meant to model the situation where, by chance, searchers had viewed all paths from non-relevant documents. I use the top-ranked sentences from the non-relevant documents to create the representations that comprise the relevance path. I use these sentences as non-relevant information and not, say the bottom-ranked sentences from non-relevant documents. This is potentially more realistic, as when used in real retrieval situations a search system implementing these techniques will always use top-ranked sentences to form document representations, regardless of whether the documents are relevant or non-relevant.

In total there were 65,826 possible path routes (i.e., 54 × 1219 [28]) for each of the six term selection models tested. The paths were again ranked based on the marginal correlation coefficient effects and the best and worst performing 10 paths chosen for this analysis. As suggested earlier in this chapter, the paths chosen from negative documents were assumed to be shorter than relevant paths. For each model, in Tables 8.6 and 8.7 I show the average path performance, the average number of terms and the proportion that are stopwords.

Table 8.6
Average best path performance in Scenario 2.

Term selection model	Rank order	Marginal Correlation	Length	Number of Terms	Indicativity	
					Document	Distribution
bvm	4	0.303	3.9	144 (45.5%)	0.258	0.010
jeff	2	0.392	3.5	165 (44.7%)	0.507	0.029
wpq.doc	1	0.434	–	–	1.000	0.025
wpq.path	6	0.239	3.7	146 (47.0%)	0.294	0.008
wpq.ost	3	0.332	3.4	139 (47.7%)	0.220	0.007
ran	5	0.244	4.0	163 (47.1%)	0.176	0.013

[28] In total, there were 1219 non-relevant documents in the top 30 retrieved for all 50 search topics used.

Table 8.7
Average worst path performance in Scenario 2.

Term selection model	Rank order	Marginal Correlation	Length	Number of Terms	Indicativity	
					Document	Distribution
bvm	3	– 0.478	3.6	150 (46.6%)	0.203	0.010
jeff	2	– 0.433	3.8	168 (46.8%)	0.388	0.027
wpq.doc	6	– 0.627	–	–	1.000	0.024
wpq.path	5	– 0.517	3.5	142 (42.7%)	0.246	0.004
wpq.ost	1	– 0.416	3.7	160 (46.3%)	0.254	0.005
ran	4	– 0.513	3.9	147 (50.3%)	0.188	0.008

The Jeffrey's Conditioning and wpq.doc models outperform the other term selection models. However, the wpq.doc model appears most variable with the highest marginal gains but also the highest losses. In a similar way to Scenario 1, the indicativity of the relevant document distribution is a good measure of the quality of the relevance path. Also, since the paths are taken from non-relevant documents the indicativity of the relevant distribution (created from relevant documents) is lower than paths from relevant documents, shown in Tables 8.6 and 8.7. Also, for paths from non-relevant documents, there appears to be no association between path performance and relevant distribution indicativity.

In Scenario 2 (as in Scenario 1), the path length, the number of terms, number of those terms that were stopwords appears to have no effect on path performance. For Scenario 1 and Scenario 2 I did not measure precision after each path. Across relevant and non-relevant documents there were 81,000 paths in total. It was not feasible to run all paths through the SMART system to determine marginal precision effects. In Scenarios 3a – 5b, I demonstrate a close relationship between the rate of learning and measures of precision; where it may not be practical to compute precision, correlation coefficients may be a reasonable approximation.

8.4.3 Scenarios 3a and 3b: Subset of Paths

The relevant subset strategy used a set of relevance paths taken from the top-ranked relevant documents. This scenario models the situation that may arise out of chance if all the information a searcher views is from documents that were relevant.

8.4.3.1 Search Effectiveness

In Scenario 3a measured search effectiveness for each of the models through their effects on precision. Figure 8.1 shows the average *11-point precision* [29] values for each model across all iterations and 10 experimental runs. As the figure illustrates, precision increases as the

[29] The average precision across 11 *recall* values ranging from 0.0 to 1.0, with an increment of 0.1.

number of iterations increases. Figure 8.1 presents the actual precision values across all 20 iterations. The Jeffrey's Conditioning and Binary Voting Models outperform the other implicit feedback models, with large increases inside the first five iterations. Both models are quick to respond to implicit relevance information, with the largest marginal increases (change from one iteration to the next) coming in the first iteration. The other models do not perform as well, but steadily increase until around 10 iterations where precision levels out.

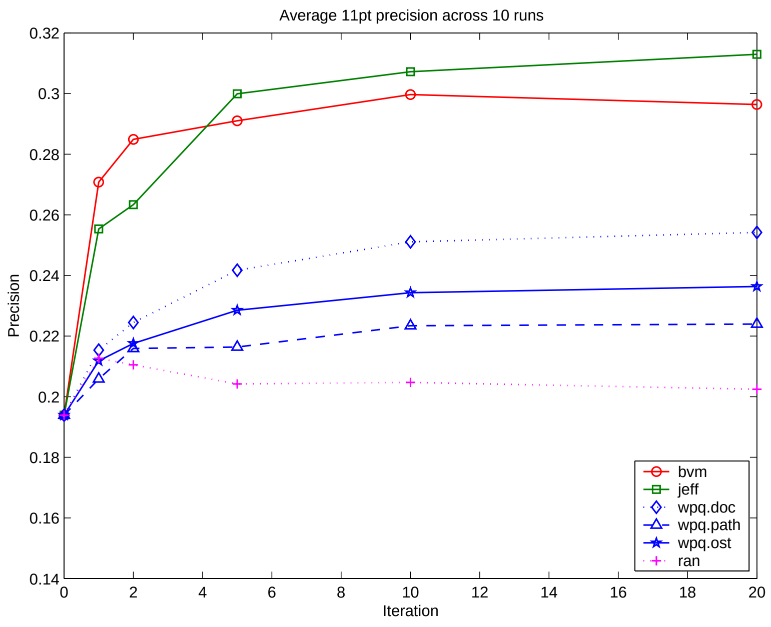


Figure 8.1. Average 11-point precision across 10 experimental runs in Scenario 3a.

Table 8.8 illustrates the marginal differences more clearly than Figure 8.1, showing the percentage change overall and the marginal percentage change at each iteration.

Table 8.8
Percentage change in precision per iteration in Scenario 3a. Overall change in first column, marginal change in second shaded column. Highest percentage in each column in bold.

Model	Iterations									
	1		2		5		10		20	
bvm	**28.4**	–	**31.9**	+ **4.9**	33.4	+ 2.9	35.3	+ 2.9	34.6	– 1.1
jeff	24.1	–	26.4	+ 3.0	**35.3**	+ **12.2**	**36.9**	+ 2.4	**38.0**	+ **1.8**
wpq.doc	10.0	–	13.6	+ 4.1	19.8	+ 7.1	22.8	+ **3.7**	23.7	+ 1.2
wpq.path	5.8	–	10.2	+ 4.6	10.4	+ 0.2	13.2	+ 3.2	13.4	+ 0.2
wpq.ost	8.5	–	10.9	+ 2.6	17.2	+ 4.8	17.2	+ 2.5	18.0	+ 0.9
ran	8.8	–	7.9	– 1.1	5.0	– 3.1	5.3	+ 0.2	4.2	– 1.1

As Table 8.8 shows, the largest increases in precision come from the Binary Voting Model and the Jeffrey's Conditioning Model. Although after 20 iterations the marginal effects of all models appear slight. The random model performs poorly, although still leads to small overall increases in precision over the baseline. Even though the random model assigned each term a random score, the paths selected by the simulation were still query-relevant. My results show that choosing terms randomly from paths can slightly improve short queries.

The *wpq*-based models appeared to follow a similar trend. At each iteration a one-way repeated measures ANOVA was carried out to compare all three *wpq*-based models and *t*-tests for pair-wise comparisons where appropriate. During the first two iterations, there were no significant differences (iteration 1: *F*(2,27) = 2.258, p = .12, iteration 2: *F*(2,27) = 1.803, p = .18) between the *wpq* models tested. ANOVAs across iterations 5, 10 and 20 suggested there were significant differences in precision between the three *wpq*-models. A series of *t*-tests revealed the *wpq document model* performed significantly better than both path-based *wpq* models (ostensive-path and path) for iterations 5, 10 and 20 (p < 0.05). The relevance paths were not of sufficient size and did not contain a sufficient mixture of terms from which *wpq* could choose candidates for query expansion.

8.4.3.2 Relevance Learning

How well the implicit models trained themselves when given relevance information by the simulation was measured. This was done through the degree of correlation between the ordered list of terms in the topic's relevant distribution and the ordered list of terms chosen by the implicit model; Figures 8.2 and 8.3 show the average Spearman and Kendall correlation coefficients across all 43 topics.

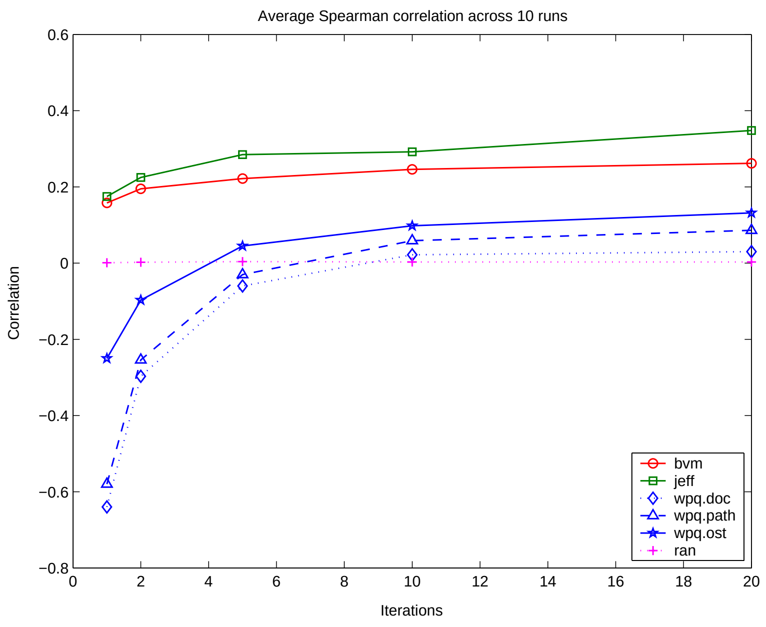


Figure 8.2. Average Spearman correlation coefficient across 10 runs in Scenario 3a.

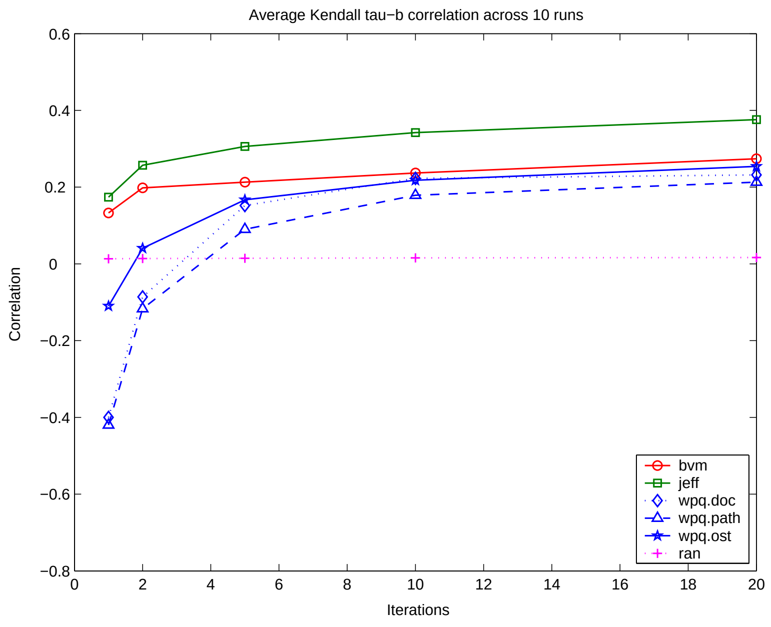


Figure 8.3. Average Kendall correlation coefficient across 10 runs in Scenario 3a.

Both coefficients follow similar trends for all implicit models. Again the Jeffrey's Conditioning and Binary Voting Model learn at a faster rate, with the Jeffrey's Conditioning Model performing best. The random model returns a coefficient value close to zero with both coefficients. In both cases a value of zero implies no correlation between the two lists, and this was to be expected if the model randomly ordered the term list. For all other models the coefficients tends to one, implying that the models were *learning* the relevant distribution from the given relevance information. Both the Jeffrey's Conditioning Model and the Binary Voting Model obtain high levels of correlation after the first iteration, whereas the *wpq* models need more *training* to reach a level where the terms they recommend appear to match those in the relevant distribution.

In Scenario 3b the paths were chosen at random from the set of paths extracted from relevant documents. However, the path length distribution was used to control the number of paths of different lengths that were used in the simulation. The results of findings of this scenario demonstrated little difference with the random paths approach used in Scenario 3a.

8.4.4 Scenarios 4a and 4b: Subset of Paths

Scenarios 4a and 4b, in a similar way to Scenarios 3a and 3b, use a subset of available paths. This scenario models the situation that may arise if, by chance, all information a searcher views is from documents that were non-relevant. It is reasonable to assume that searchers will view *some* information from non-relevant documents as they search. It is only in extreme situations where *all* the information they view is from non-relevant documents. These scenarios model such an extreme situation.

8.4.4.1 Search Effectiveness

I measured search effectiveness for each of the models through their effects on precision. Figure 8.4 shows the 11-point precision values for each model across all 20 iterations. All models increased the precision after the first iteration, however as the figure illustrates, some models increased overall precision and some reduced overall precision.

The Jeffrey's Conditioning and Binary Voting Models outperform the other implicit feedback models. Although the increases in precision are small, the Jeffrey's Conditioning and Binary Voting Models seem better able to create effective search queries in situations where relevant information is difficult to find. That is, they seem better able to use paths from non-relevant documents to select terms for query modification. The other models do not perform as well, but steadily increase until around 10 iterations where precision levels out.

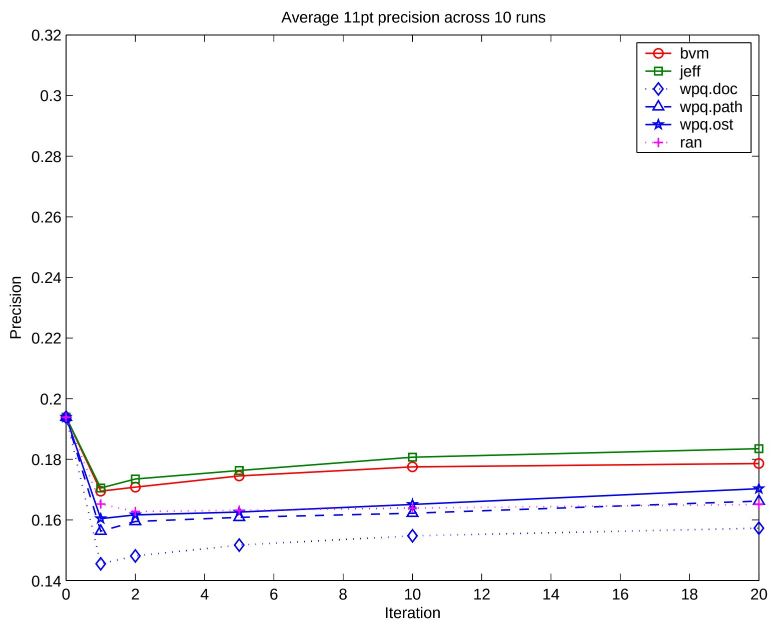


Figure 8.4. Average 11-point precision across 10 experimental runs in Scenario 4a.

The paths from non-relevant documents typically contain very few or no query terms. The relevance paths are sentence-based and sentences are scored based on the algorithm for scoring Top-Ranking Sentences described in Chapter Three. A large proportion of each sentence's score is derived from its relation to the query. If there are few query terms, then other factors, such as the location of a sentence in a document and any titles in documents that also appear in the document title are used to weight relevance paths. The paths chosen are therefore document-dependent, not query-dependent and may cover a number of unrelated themes. Whilst all models appear to be affected by the presence of non-relevant information the Jeffrey's Conditioning and Binary Voting Models appear most able to operate most effectively. The difference between all models was not significant with ANOVA across any iterations (*F*(5,54) = 1.844, p = .120). Over time all models increase precision slightly. With the exception of the *wpq.doc* model all models take terms from relevance paths that extract the most potentially useful parts of documents. Whilst the documents were classified by the TREC assessors as non-relevant they had some features that made the SMART system rank them higher than other documents in the collection. They may contain additional words that could be of use in creating enhanced search queries.

Table 8.9 illustrates the marginal difference more clearly than Figure 8.5, showing the percentage change overall and the marginal percentage change at each iteration.

Table 8.9
Percentage change in precision per iteration in Scenario 4a. Overall change in first column, marginal change in second shaded column. Highest percentage in each column in bold.

	Iterations									
Model	1		2		5		10		20	
bvm	−14.4	–	−13.5	+ 0.8	−11.1	+ 2.1	− 9.2	+ 1.7	− 8.6	+ 0.6
jeff	**−13.7**	–	**−11.8**	+ 1.8	**−10.0**	+ 1.6	**− 7.3**	+ **2.4**	**− 5.7**	+ 1.5
wpq.doc	−33.3	–	−30.9	+ 1.8	−27.8	+ **2.4**	−25.3	+ 2.0	−23.3	+ 1.6
wpq.path	−24.0	–	−21.6	+ **2.0**	−20.5	+ 0.8	−19.5	+ 0.8	−16.7	+ **2.4**
wpq.ost	−20.9	–	−20.0	+ 0.7	−19.2	+ 0.6	−17.4	+ 1.5	−13.9	+ 0.3
ran	−17.3	–	−19.1	− 1.3	−18.8	+ 0.3	−18.3	+ 0.4	−17.6	+ 0.7

It should be noted that using linear regression there is no significant difference in the rate of learning in all models *after the first iteration* (*all* $r^2 \geq .8941$ and *all* $t(38) \geq 17.91$, $p \leq .05$). As was demonstrated in Scenarios 3a and 3b, the Jeffrey's Conditioning and Binary Voting Models perform better than the other models in the first iteration. When presented with paths from non-relevant documents these models seem better able to extract useful terms. As is shown in Table 8.9, it is the first iteration that provides the overall increase in precision; after iteration one the marginal changes are similar for all models.

8.4.4.2 Relevance Learning

I measured how well the implicit models trained themselves when given relevance information by the simulation. The relevance learning trend of the models was similar to

Scenario 3, and was measured in the same way; Figures 8.5 and 8.6 shows the average Spearman and Kendall correlation coefficients across all 50 topics.

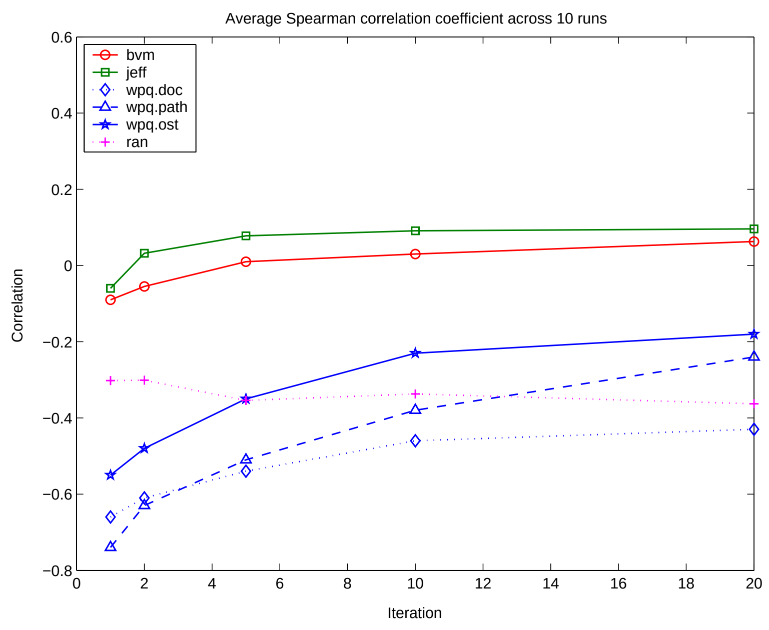


Figure 8.5. Average Spearman correlation coefficient across 10 runs in Scenario 4a.

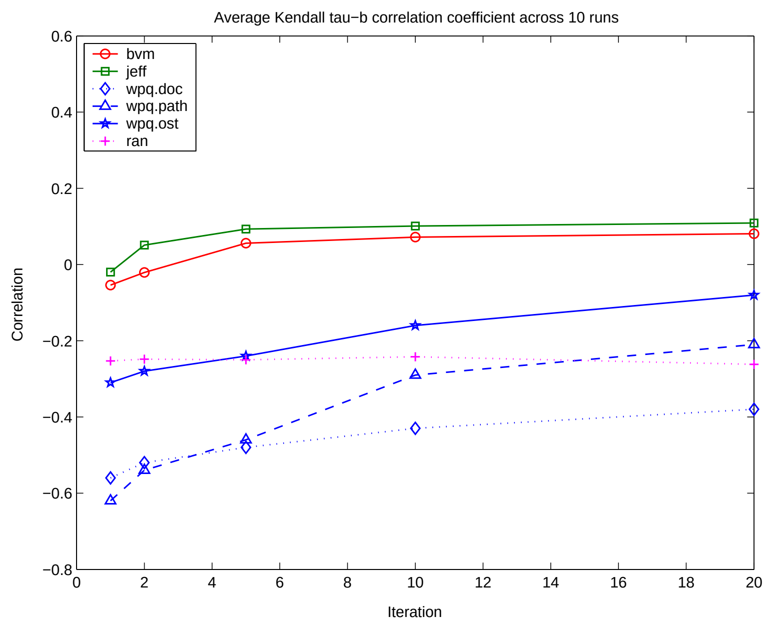


Figure 8.6. Average Kendall correlation coefficient across 10 runs in Scenario 4a.

The results show that in a similar way to Scenario 3, the models learn over time. However, since they are being shown information from non-relevant documents they do not learn the relevant distribution (composed of relevant documents) at as fast a rate and do not finish with as high a correlation as in Scenarios 3a and 3b. The random model returns a coefficient value close to zero with both coefficients in 3a and 3b. However, in this scenario it is lower, suggesting it starts at a low rate of learning and does not improve on this. The models based on *wpq* also perform poorly initially but improve gradually as the search proceeds.

In a similar way to 3b, Scenario 4b revealed only a slight difference between the selection of paths randomly (as in 4a) and the use of the path length distributions. When paths were selected randomly there was a restriction on their length, which could not exceed three steps. When the path length distributions were used some paths were allowed to exceed this three step threshold, meaning the system was presented with more information. However, since this information was from irrelevant documents it had a detrimental effect on the performance of all models and led to slightly larger reductions in search effectiveness.

8.4.5 Scenarios 5a and 5b: Related Paths

This scenario uses the 'Related Paths' approach described in Section 8.3.3.2.1 to select paths from relevant and non-relevant documents. Search effectiveness (monitored through precision) and relevance learning (measured through correlation coefficients) are monitored for different levels of wandering. In this section I summarise the findings and present the average for all levels of wandering (i.e., the average for wandering levels at 10, 20, 30, 40 and 50%). I present the actual values obtained for each of these levels in Appendix A. This approach is potentially more realistic than the experimental scenarios presented so far in this chapter, as it is conceivable that searchers will view irrelevant information as they search.

8.4.5.1 Search Effectiveness

As in previous scenarios the 11-point precision value was measured at iterations 1, 2, 5, 10 and 20. In Figure 8.7 I present the average precision value across all 10 runs and across all levels of wandering. The trend is the same as in earlier scenarios, with the Jeffrey's Conditioning and Binary Voting Models leading to overall increases in precision. However, because I introduce non-relevant 'noise' into the calculation, the overall increases in precision are not as large as in Scenarios 3a and 3b.

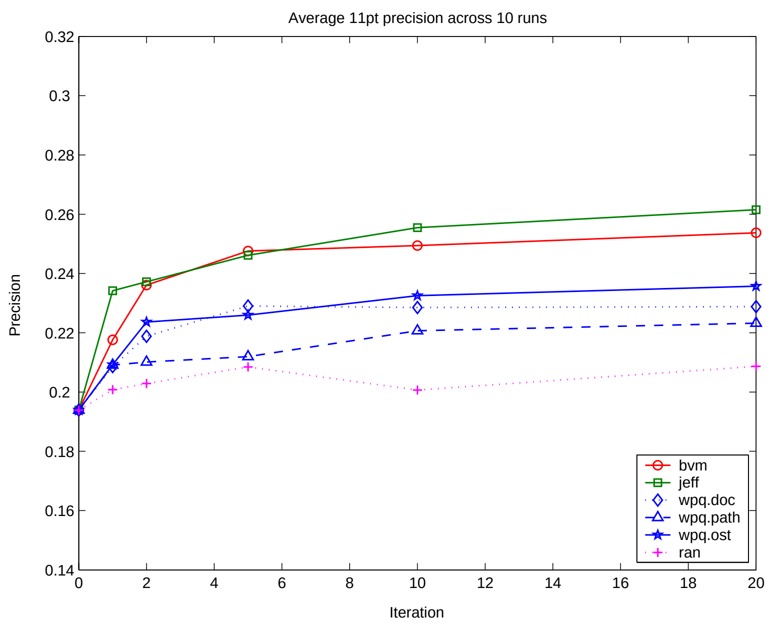


Figure 8.7. Average 11-point precision across 10 experimental runs in Scenario 5a.

The percentage change in overall and marginal precision for each of the models is shown in Table 8.10.

Table 8.10
Percentage change in precision per iteration in Scenario 5a. Overall change in first column, marginal change in second shaded column. Highest percentage in each column in bold.

Model	Iterations									
	1		2		5		10		20	
bvm	10.9	–	17.9	+ **7.8**	**21.7**	+ **4.6**	22.3	+ 0.7	23.6	+ 1.7
jeff	**17.2**	–	**18.3**	+ 1.3	21.2	+ 3.6	**24.1**	+ 3.6	**25.9**	+ 2.3
wpq.doc	7.0	–	11.4	+ 4.7	15.3	+ 4.5	15.1	– 0.2	15.3	+ 0.1
wpq.path	7.3	–	7.7	+ 0.5	8.5	+ 0.9	12.1	+ **3.9**	13.1	+ 1.1
wpq.ost	7.3	–	13.3	+ 6.4	14.2	+ 1.0	16.6	+ 2.8	17.7	+ 1.4
ran	3.4	–	4.4	+ 1.0	7.0	+ 2.7	3.4	– 3.9	7.1	+ **3.9**

As the level of wandering rises, increases in the level of precision drop. Viewing information from non-relevant documents (as Scenarios 4a and 4b demonstrate) is to reduce the overall effectiveness of all the term selection models. Nonetheless, the Jeffrey's Conditioning and Binary Voting Models still outperform the others.

8.4.5.2 Relevance Learning

The models' ability to improve their understanding of what information is relevant was again measured using the Spearman and Kendall correlation coefficients. The values for both coefficients at iterations 1, 2, 5, 10 and 20 are presented in Figures 8.8 and 8.9 respectively.

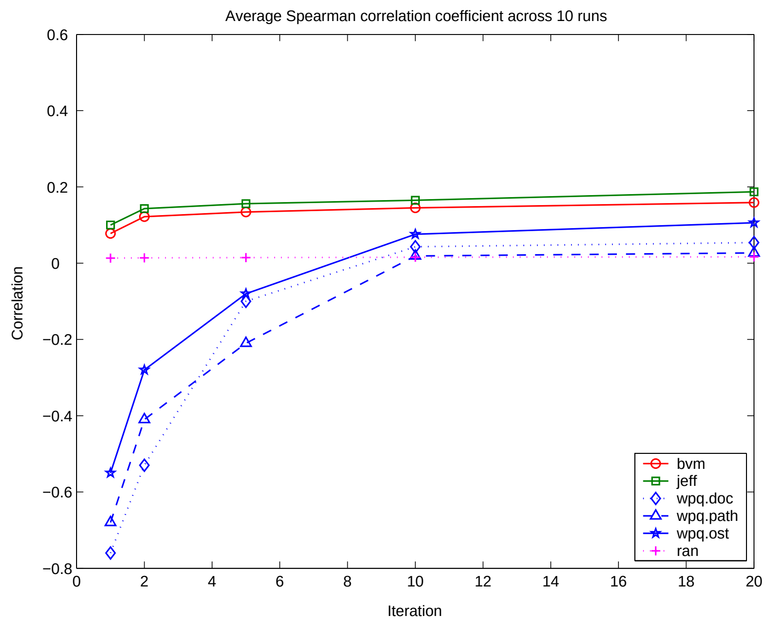


Figure 8.8. Average Spearman correlation coefficient across 10 runs in Scenario 5a.

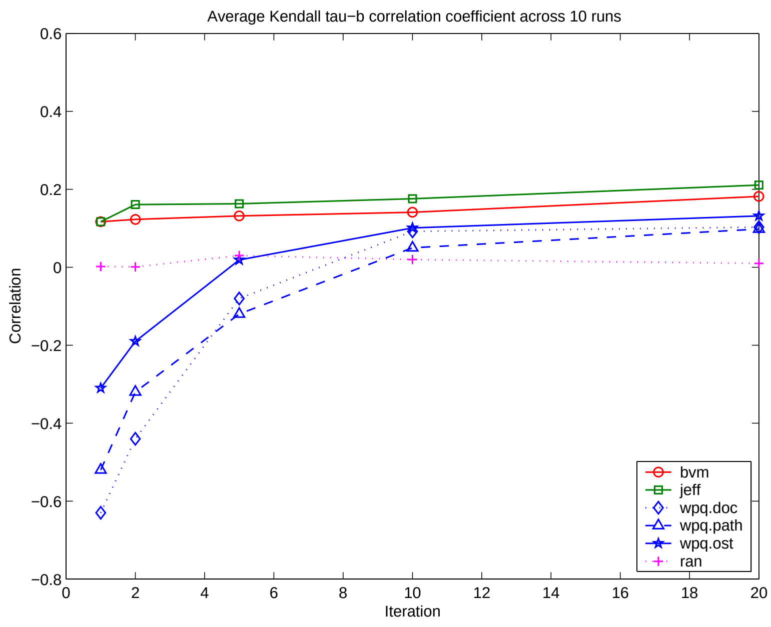


Figure 8.9. Average Kendall correlation coefficient across 10 runs in Scenario 5a.

Even though the models are shown potentially non-relevant information the results still demonstrate that the models are able to learn. However, their ability to do so is affected by the level of wandering. As wandering increases the rate at which the models learn relevance decreases. The actual correlation values for different levels of wandering are presented in Appendix B.

In Scenario 5b, where path length distributions restricted the length of visited paths there were slight differences with this scenario. The restrictions imposed meant that the simulation had to choose paths that may not be as similar to the current path as other candidate paths, but had to be chosen to full the percentage quota of the distribution. The overall effectiveness of the models was reduced by around 5% by imposing the path length restriction. I present the actual values for Scenario 5b in Appendix C. In the next section I discuss this study's results.

8.5 Discussion

The implicit feedback models evaluated in this paper *all* increased search effectiveness through query modification. However, two models performed particularly well; the Jeffrey's Conditioning Model and the Binary Voting Model. Both models improved precision and developed lists of terms that were closely correlated to those of the relevant distribution.

Initially, in most scenarios, the Jeffrey's Conditioning Model does not perform as well as the Binary Voting Model at the start of the search. However, after five paths it creates more effective queries and from then on performs increasingly better than it. The Jeffrey's Conditioning Model uses prior evidence that is independent of the searcher's interaction. Initial decisions are made based on this prior evidence, and for the first few iterations it is reasonable to assume that this evidence still plays a part in term selection. However, as more evidence is gathered from searcher interaction the terms selected by the Jeffrey's Conditioning Model improve.

An advantage of the Binary Voting Model, and perhaps why it performs well in the initial stages is that it does not rely on any prior evidence, selecting terms based only on the representations viewed by the searcher. However, the lists of potential terms offered stagnates after 10 paths, since in the Binary Voting Model the effect of the scoring is cumulative, the high-scoring, high-occurrence terms, obtain a higher score after only a few initial paths and cannot be succeeded by lower-ranked terms in later paths. This often means that the same query is presented in iterations 10 and 20.

The implicit feedback models learned relevance from the evidence provided to them by the simulation. This form of reinforcement learning (Mitchell, 1997), where the model was repeatedly shown examples of relevant information, allowed me to test how well each model trained itself to recognise relevance. From the six models tested, the findings showed that the Jeffrey's Conditioning and Binary Voting Models learned at the fastest rate. In the first few iterations those models based on *wpq* performed poorly in all retrieval scenarios, suggesting that these models need more training to reach an acceptable level of relevance recognition and that the Jeffrey's Conditioning and Binary Voting Models make a more efficient use of relevance information. Linear regression was used and compared the *rate of learning* against *precision* for each of the six implicit feedback models. The results showed that for all models, the rate of learning (i.e., *Spearman's rho* and *Kendall's tau*) followed the same trend as precision (*all* $r^2 \geq .8154$ and *all* $t(38) \geq 5.34$, $p \leq .05$). The rate in which the models learn relevance appears to match the rate in which they are able to improve search effectiveness.

The findings of the study show that the Jeffrey's Conditioning and Binary Voting Models are able to perform more effectively than the baselines when all the paths presented to them are from non-relevant documents (Scenarios 4a and 4b) and only a proportion of the paths are (Scenarios 5a and 5b). Whilst it is understandable that models can perform effectively when shown only relevant information, it is important for them to also perform well in situations where non-relevant information is also shown. This is important in implicit feedback models as they assume a degree of relevance in all the information searchers view.

From the three models that implement different versions of the *wpq* algorithm, the wpq.doc model performed best for all relevant documents (Scenarios 3a and 3b) and worst for all non-relevant documents (Scenarios 4a and 4b). This model is more sensitive to the relevance of documents used than the path-based models. The document model must use all of the content of each document, whereas relevance paths comprise only the potentially useful parts of documents and hence reduce the likelihood that erroneous terms are selected. Since documents will typically be longer than relevance paths, the contribution a single document makes to term scoring may typically exceed that of one relevance path.

In this study I have also shown that paths that lead to largest marginal increases in relevance learning are those that are indicative of the term distribution they are trying to learn. That is, paths that are indicative of the terms that occur over all relevant documents are likely to be high quality paths. There is no relationship between the number of steps in a path, the number of tokens in a path, or the percentage of stopwords in a path and the overall effectiveness of a path. Therefore, it is not how many words a path contains that determines the effectiveness of a relevance path, but what those words are, and how those words are distributed in the set of relevant documents.

For almost all iterations on all models, the marginal increases in precision and correlation reduce as more relevant information is presented. The models appear to reach a point of saturation at around 10 paths, where the benefits of showing 10 more paths (i.e., going to iteration 20) are only very slight and are perhaps outweighed by the costs of further interaction. It is perhaps at this point where searcher needs would be best served with a new injection of different information or explicit searcher involvement.

Simulation-based techniques of this nature can be useful for designers of search systems who can more fully test the suitability of implicit feedback models to the interface design and modify the models or interfaces where appropriate. In the next section I summarise this chapter.

8.6 Chapter Summary

In this chapter a simulation-based evaluation methodology called SIMPLE was presented and used to evaluate a variety of implicit feedback models. The models under test were ostensive in nature and use the exploration of the information space and the viewing of information as an indication of relevance. Six models in total were tested, each employing a different term selection stratagem.

The simulated approach used to test the models assumed the role of a searcher 'viewing' relevant documents and relevance paths between granular representations of documents. The simulation passes the information it viewed to the implicit feedback models, which use this evidence to select terms to best describe this information. I investigated the degree to which each of the models improved search effectiveness and learned relevance. From the six models tested, the Jeffrey's Conditioning Model provided the highest levels of precision and the highest rate of learning.

Simulation experiments are a reasonable way to test the worth of implicit feedback models such as those presented in this chapter. However, whilst the simulation allowed me to benchmark model performance, evaluation with simulations is only formative and there is a need for further investigation of the best performing model when it is employed by real searchers engaged in IIR. In Part IV a user experiment is conducted of feedback systems that use the Jeffrey's Conditioning Model for term selection. In this chapter I have assessed the performance of the model objectively, using measures of search effectiveness and relevance learning. In the experiment in subsequent chapters, the performance of the model is assessed using human subjects.