Extending Probabilistic Data Fusion Using Sliding Windows

David Lillis[1], Fergus Toolan[2], Rem Collier[1], and John Dunnion[1]

[1] School of Computer Science and Informatics
University College Dublin
{david.lillis, rem.collier, john.dunnion}@ucd.ie
[2] Department of Computing Science
Griffith College Dublin
fergus.toolan@gcd.ie

Abstract. Recent developments in the field of data fusion have seen a focus on techniques that use training queries to estimate the probability that various documents are relevant to a given query and use that information to assign scores to those documents on which they are subsequently ranked. This paper introduces SlideFuse, which builds on these techniques, introducing a sliding window in order to compensate for situations where little relevance information is available to aid in the estimation of probabilities.

SlideFuse is shown to perform favourably in comparison with CombMNZ, ProbFuse and SegFuse. CombMNZ is the standard baseline technique against which data fusion algorithms are compared whereas ProbFuse and SegFuse represent the state-of-the-art for probabilistic data fusion methods.

1 Introduction

The aim of any Information Retrieval (IR) system is the identification of documents that best satisfy a user's information need, typically expressed in terms of a textual query. Traditional approaches to IR employ algorithms responsible for analysing the contents of the documents themselves in order to return those that most closely relate to the query provided.

More recently, there is a growing body of research focused on combining the output of several such systems with the aim of creating a single set of results that will have greater relevance than the output of any individual system [1–3]. Algorithms to perform this type of combination vary according to the situations in which they are intended to be used. This paper concentrates on the "data fusion" family of algorithms, which are intended for use in cases where each input system has access to the same document collections [4]. This is distinct from "collection fusion" [5], where the document collections are disjoint, or cases where only partial overlap exists between collections.

The principal difference between these situations is that data fusion algorithms may consider the presence of a document in multiple result sets as evidence of relevance, since a document's absence in a result set can only be as a

result of it not being considered relevant by the corresponding input system. In contrast, where the overlap between document collections is not complete, the absence of a document from a result set may merely reflect its absence from the underlying document collection and so is not necessarily a reliable indication that the document has been considered to be nonrelevant.

This paper introduces SlideFuse, a novel probabilistic data fusion algorithm that uses the past performance of its underlying input systems as an indication of the probability that certain documents will be relevant to future queries. This assumption has been previously demonstrated to achieve favourable results [6, 7]. It is robust in the face of incomplete training data by utilising information about a document's neighbours as evidence of its likelihood of relevance. It does this while avoiding some of the shortfalls of existing probabilistic methods.

Section 2 gives a brief outline of previous research in the area of data fusion. In Section 3, we present an overview of how SlideFuse operates, followed by a formal definition of the algorithm in Section 4. Section 5 details the setup of the experiments that were run to evaluate the SlideFuse algorithm, the results of which are presented in Section 6. This includes a comparison with the CombMNZ algorithm, which is a standard baseline frequently used in data fusion research, as well as ProbFuse and SegFuse, two recent probabilistic data fusion techniques. Finally, conclusions and future work are discussed in Section 7.

2 Data Fusion

Traditionally, data fusion techniques fall into two broad categories: score-based fusion and rank-based fusion. Score-based techniques make use of the scores each input system uses to rank the documents in its result set. This typically necessitates the use of some form of score normalisation [8], in order to ensure that the results cannot be skewed by the use of different methods of allocating scores (e.g. one input system may score documents on a scale of 0-100 whereas another may use a scale of 0-1).

A popular approach to score-based fusion is the use of a Linear Combination [1, 9, 3]. Here, weights are attached to each input system, which are multipled by the ranking scores assigned each document. The final score for each document is the sum of these. Normalised scores have also been used in this context [10].

An important suite of data fusion techniques based on normalised scores was proposed in [8]. Of these, CombMNZ has become the standard data fusion technique against which new algorithms are compared [2, 11]. Here, the final score assigned to each document is the sum of the normalised scores it is given in each input result set, multiplied by the number of input systems that returned it. Significant work was carried out by Lee to demonstrate CombMNZ's effectiveness [12].

Interleaving is perhaps the simplest rank-based fusion technique [5]. This involves removing the top document from each input result set in turn and adding it to the fused set to be returned. Weighted variations on this have also been proposed so as to benefit input systems that have achieved superior

performance in the past [13]. Two voting-based techniques based on document ranks were proposed by Aslam and Montague [11, 14]. These used the analogy of the input systems representing few electors and the documents representing many candidates to be ranked.

An algorithm making use of the textual contents of the documents was presented in [15, 16]. Another relies on the input systems providing metadata relating to the documents they return, which can be used in the fusion process [17].

In recent times, a variation of rank-based fusion has emerged, whereby result sets are divided into segments and documents are assigned a score based on the segments in which they appear, rather than their exact rank within the result set. The ProbFuse algorithm [6, 18] divides each result set into equal length segments and uses training data to estimate the probability that a document returned in a particular segment by a particular input system is relevant. This is done by calculating the proportion of documents returned in each segment by each input system that are relevant to the training queries, compared to nonrelevant documents.

A similar approach is taken with SegFuse [7], with the major exception being that the segments are not of equal length, but rather increase in size exponentially later in the result set. As relevant documents are most likely to occur in the early part of a result set, maintaining small segment sizes in early positions advantages these early documents, as they are less likely to be grouped with less relevant documents occurring later on. SegFuse also takes normalised scores into account.

3   SlideFuse: Introduction

Existing segment-based data fusion techniques ProbFuse and SegFuse use the probability that a document is relevant to assign a score on which it is eventually ranked in the final result set. This probability is estimated by analysing the results of a number of training queries for which relevance judgments are available. Relevance judgments are typically included with IR test collections, and specify which documents in the collection have been judged to be relevant, or nonrelevant, to test queries. However, with large document collections these judgments tend to be incomplete, meaning that only relatively few documents have been judged for each query, leaving the majority unjudged. This incompleteness causes difficulty in analysing training data, as there may be positions in result sets in which a document that is known to be relevant is never returned, though this does not necessarily entail that a relevant document is never located at that rank.

For this reason, calculating probabilities at the individual rank level results in an extremely jagged probability distribution. For instance, with the Web Track from the TREC-2004 conference (which is the document collection used in the experiments presented in Section 5), calculating the probability for each position results in the graph presented in Figure 1. In that figure, the probability value

used in each position is the number of relevant documents returned in that postion over all the training queries, divided by the total number of training queries that returned a document in that position (i.e. a result set of only 100 documents in length will not have returned a document in position 101).

One motivation behind segmenting result sets is to counter this effect, by not estimating the probability of relevance of a document returned at a particular rank solely based on documents returned at that exact rank for the training queries. Instead, relevant documents returned at other positions within the same segment are also taken into account, so smoothing the distribution of probability scores.

One consequence of this approach is that it is possible for a significant drop in probability score to occur at the boundary between segments. This effect is illustrated in Figure 2. For example, in a result set divided into segments of 40 documents each, the probability associated with the document returned in position 40 is likely to be much higher than that of the document returned in position 41. This is because the probability for the segment containing position 40 is calculated using positions 1 through 40, whereas the segment containing position 41 ranges from position 41 to position 80. As the former encompasses documents much higher in the result set (that are more likely to be relevant than documents further down the result set), position 40 is given an artificial advantage over position 41. This is easily demonstrated by plotting a graph of probability score against position. Unlike ProbFuse, SegFuse changes the size of each segment in different areas of the result set, with the smallest segments being at the beginning. This has the effect of reducing the distance between such segment boundaries at the beginning of the result set, where relevant documents are most likely to appear and consequently reducing the occurrence of sudden changes in probability scores.
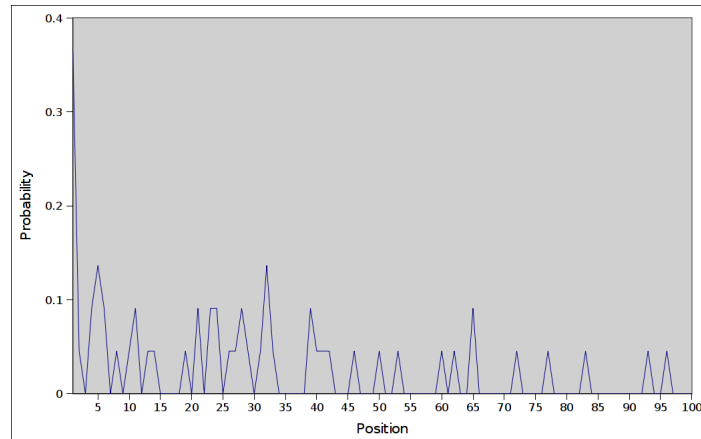

Fig. 1. Probability Distribution using Individual Positions

In order to address the problem of the sudden drops in the probability scores associated with segmentation, and the problem of incomplete relevance judgments, we introduce the concept of a window surrounding each rank, where the probability assigned to that rank is based on the proportion of relevant documents located in its surrounding window during the training phase. For example, if we define the size of the sliding window to extend to 5 documents on either side of the relevant position, the window for rank 40 will extend from position 35 to position 45 inclusive. Similarly, the sliding window for position 41 extends from rank 36 to rank 46. With this approach, the problem of the location of segment boundaries is eliminated, as it is the closest neighbouring positions that
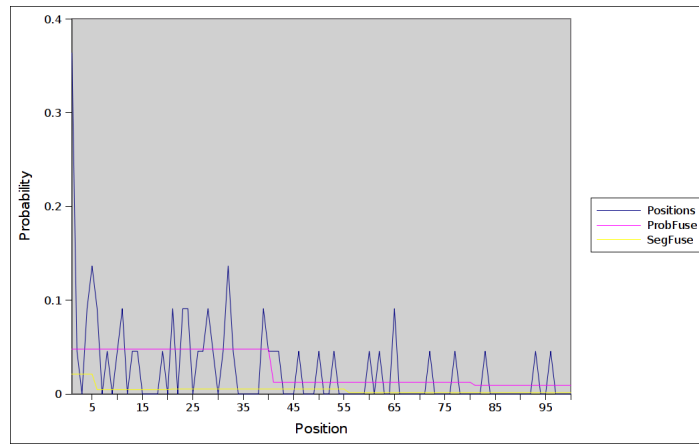

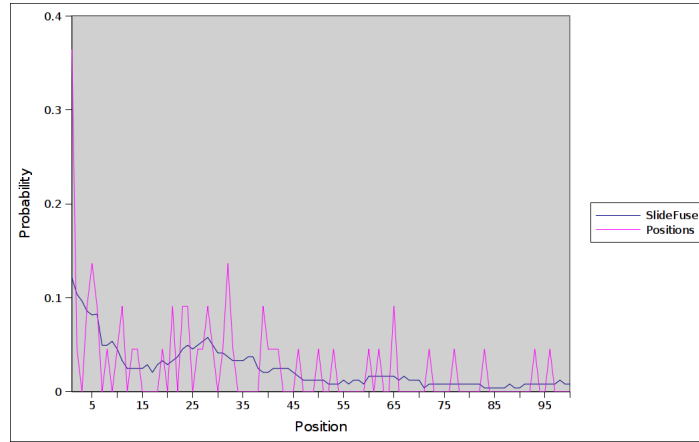**Fig. 2.** Probability Distribution using ProbFuse and SegFuse

Fig. 3. Probability Distribution using SlideFuse

are always taken into account. The effect on probability distribution is shown in Figure 3. As a special case, SlideFuse ensures that a sliding window cannot extend beyond the boundaries of the result set.

The use of training data entails that the scores that are ultimately assigned to each document are based on the past performance of each input system, thus encompassing an implied weighting system wereby documents returned by input systems with a prior record of greater effectiveness will receive a higher score. A formal mathematical description is presented in Section 4.

4 SlideFuse: Description

In common with other probabilistic data fusion techniques, SlideFuse requires both a training phase and a fusion phase. In the training phase, relevance information is gleaned from result sets returned in response to training queries for which relevance judgments are available. Later, this training data is used to fuse result sets produced by the same input systems relating to other queries.

4.1 Training Phase: Rank Probability

The training phase consists of estimating for each input system the probability that a document returned in any given rank in that system's result set is relevant.

Formally, $P(d_p|s)$, the probability that a document d returned in position p of a result set is relevant, given that is has been returned by input system s is given by

$$P(d_p|s) = \frac{\sum_{q \in Q_p} R_{d_p,q}}{Q_p} \tag{1}$$

where Q_p is the set of all training queries for which at least p documents were returned by the input system and $R_{d_p,q}$ is the relevance of the document d_p to query q (1 if the document is relevant, 0 if not). This is calculated for each input system to be used in the fusion phase.

4.2 Fusion Phase: Window Boundaries

As noted in Section 3, using the probability at each rank leads to inconsistent results on document collections with incomplete relevance judgments, due to the high number of documents in each result set that have not been judged relevant (and are therefore assumed not to be relevant). In order to achieve more useful probability values, we construct a window around each position, so as to make use of relevance information about near neighbours when assigning probabilities to individual ranks.

The start and end points (a and b respectively) of the sliding window surrounding each result set position p are given by

$$a = \begin{cases} p - w & p - w >= 0 \\ 0 & p - w < 0 \end{cases} \tag{2}$$

$$b = \begin{cases} p + w & p + w < N \\ N - 1 & p + w >= N \end{cases} \tag{3}$$

where w is a parameter that indicates how many positions on either side of p should be included in the window and N is the total number of documents in the result set. In effect, the above definitions of a and b ensure that the window cannot begin before the first document in the result set and also cannot extend beyond the last document.

4.3 Fusion Phase: Assigning Probabilities to Windows

Once the window boundaries have been set around each position of each of the result sets that are to be fused, the next stage in the fusion process is to assign a probability score to each position based on those positions contained in the window surrounding it.

$P(d_{p,w}|s)$, the probability of relevance of document d in position p using a window size of w documents either side of p, given that it has been returned by input system s is given by

$$P(d_{p,w}|s) = \frac{\sum_{i=a}^{b} P(d_i|s)}{b - a + 1} \tag{4}$$

The use of the sliding window results in a smoother decrease in the probabilities later in the result set, when compared with using probabilities based on data available at each position alone.

4.4 Fusion Phase: Ranking Score

Once the above stages have been completed, the final step is to assign a score to each document. R_d, the final ranking score given to document d is given by

$$R_d = \sum_{s \in S} P(d_{p,w}|s) \tag{5}$$

where S is the set of all input systems used and p is the position in which document d was returned by input system s. Using the sum of the probability scores makes use of the "Chorus Effect", which argues that multiple input systems agreeing on the relevance of a document is evidence that the document in question is actually relevant [3]. The "Skimming Effect" is also important in the context of data fusion [3]. This states that since relevant documents are most likely to be located in early positions in a result set, weighting highly-ranked documents heavily is beneficial when performing fusion. Although there is no explicit consideration of this effect made in the definition of SlideFuse, the probability distribution in Figure 3 shows that this increased likelihood of relevance in early positions automatically benefits these highly-ranked documents.

5 Experiment Setup

The document collection used for evaluation is the Web Track from the TREC-2004 conference [19]. A feature of this document collection is that the relevance judgments are extremely incomplete. The available data includes 74 topfiles (each containing result sets produced by a single input system in response to each of 225 queries). A number of measures were taken in order to reduce the possibility of any bias being introduced by either the selection of input systems or the ordering of the queries.

Five runs of the experiment were performed. For each run, six topfiles were selected and the result sets from those topfiles were fused using SlideFuse, ProbFuse, SegFuse and CombMNZ. No topfile was used in more than one experimental run, the result of which being that of the 64 topfiles available, 30 were used for the purposes of this experiment. So as to eliminate the possibility of the ordering of the queries introducing any sort of bias, each run was performed five times, with the queries being shuffled each time. After shuffling, the first 10% of queries were used for the purposes of training SlideFuse, ProbFuse and SegFuse. As the CombMNZ algorithm does not require a training phase, these training queries were ignored for that technique. The evaluation results presented below for each run are the average evaluation results from all of the various query orderings.

When running ProbFuse, each result set was divided into 25 segments, as in [6]. For the purposes of SlideFuse, the value of the w parameter was set to 5 (i.e. 5 documents on both sides of each position were included in the window). It is desirable to use a small value for w, in order that the probabilities at each position are only influenced by positions that are close by. However, initial experiments showed that using windows that are too small failed to fully address the problem outlined in Section 3, as there were still positions for which probabilities could not be calculated due to a lack of available relevance judgments.

When performing the evaluation of the four data fusion techniques, three evaluation measures were used: *Mean Average Precision (MAP)* is the mean of the precision scores obtained after each relevant document has been retrieved. Relevant documents that are not included in the result set are given a precision of zero. MAP assumes that documents that have not been judged are nonrelevant. The *bpref* measure evaluates the relative position of relevant and nonrelevant documents, ignoring documents that are unjudged. It was proposed by Buckley and Voorhees to cater for situations where relevance judgments are incomplete [20]. *P10* measures the precision after 10 documents have been returned. Research has demonstrated that the vast majority of users of IR systems only examine the top 10 documents presented to them [21]. Thus, the P10 measure places emphasis on documents returned in those positions where they are likely to be of use to the user.

Table 1 illustrates the results of initial experiments aimed at choosing an appropriate training set size. Fusion was performed using each algorithm, with the training set sizes set to 10%, 20%, 30%, 40% and 50% of available queries in turn. The performance of each algorithm was evaluated for each training set

size using MAP. The Coefficient of Variation relating to these scores was then calculated for each algorithm for each run. This reflects the degree to which fusion performance is affected by changing the training set size. As Table 1 illustrates, altering the number of training queries did not have any substantial effect on the performance of any of the fusion algorithms. Similar results were obtained for the bpref and P10 evaluation measures. Using only 10% of the available queries for training thus reduces the amount of training data without adversely affecting performance.

Table 1. Coefficient of Variation for MAP scores using training set sizes of 10%, 20%, 30%, 40%, 50%

	CombMNZ	ProbFuse	SegFuse	SlideFuse
first	0.0033	0.0131	0.0056	0.0056
second	0.0056	0.1188	0.0581	0.0104
third	0.0380	0.0168	0.0120	0.0103
fourth	0.0034	0.0143	0.0111	0.0019
fifth	0.0246	0.0229	0.0491	0.0179

6 Analysis of Results

The results of comparing SlideFuse with CombMNZ, ProbFuse and SegFuse are shown in Tables 2, 3 and 4. Each table presents the results from each of the five runs, along with the average result for each fusion technique. The "vs. Best" column displays the percentage difference between SlideFuse and the best of the other techniques (which is highlighted in bold in each case). The average in that column is the percentage difference between the average SlideFuse score and the best average score amongst the other algorithms. Values marked with "*" are statistically significant for a significance level of 5%, using a paired t-test. Entries marked with "**" are significant for a significance level of 1%.

Table 2. TREC-2004 performance of five individual runs evaluated with MAP

	CombMNZ	ProbFuse	SegFuse	SlideFuse	vs. Best
first	0.1598	**0.4045**	0.1789	0.4977	23.05% **
second	0.0783	**0.2809**	0.1493	0.4905	74.58% **
third	0.0426	0.2454	**0.4946**	0.5103	3.17% **
fourth	0.2454	0.2505	**0.4995**	0.5025	0.61%
fifth	0.1334	0.2892	**0.3348**	0.3849	14.98% **
average	0.1319	0.2941	**0.3314**	0.4772	43.99%

Table 3. TREC-2004 performance of five individual runs evaluated with bpref

	CombMNZ	ProbFuse	SegFuse	SlideFuse	vs. Best
first	0.2176	**0.2997**	0.2547	0.4009	33.75% **
second	0.3155	0.1877	**0.3529**	0.4085	15.75% **
third	0.1665	0.1281	**0.4228**	0.4331	2.44% *
fourth	0.4015	0.1375	**0.4155**	0.4131	-0.58%
fifth	0.1945	0.1968	**0.2971**	0.2996	0.83%
average	0.2591	0.1900	**0.3486**	0.3910	12.17%

Table 4. TREC-2004 performance of five individual runs evaluated with P10

	CombMNZ	ProbFuse	SegFuse	SlideFuse	vs. Best
first	0.1123	**0.1344**	0.1195	0.1413	5.15% **
second	0.0349	**0.1023**	0.0800	0.1436	40.39% **
third	0.0257	0.1164	**0.1401**	0.1445	3.15% **
fourth	0.1101	0.1124	**0.1381**	0.1408	1.96% *
fifth	0.0561	0.1070	**0.1113**	0.1189	6.83% **
average	0.0678	0.1145	**0.1178**	0.1378	16.99%

Of the baseline techniques, ProbFuse performs best on the "first" and "second" runs, with SegFuse achieving superior performance on the others, with one exception in the bpref data. Overall, SlideFuse achieves the highest evaluation scores on average for all evaluation measures, with the single exception of the bpref score for the "fourth" run where the difference is 0.58%, although this difference is not significant. Tests show that the performance improvements are statistically significant in most cases, with the exceptions being the "fourth" run when evaluated using MAP and the "fifth" run when evaluated with bpref.

Additionally, SlideFuse outperforms the best other technique in all runs using all three evaluation measures. When compared on an overall basis against any individual technique, the improvement is over 12% in all cases, and is above 40% when measured using MAP.

7 Conclusions and Future Work

This paper describes SlideFuse, a probabilistic data fusion algorithm that addresses some of the limitations of existing segment-based probabilistic techniques. On experiments using the TREC-2004 Web Track dataset, SlideFuse was shown to outperform the CombMNZ, ProbFuse and SegFuse data fusion techniques when evaluated using MAP, bpref and P10. Despite the fact that the training data available for the dataset is incomplete, SlideFuse was still capable of outperforming two algorithms that use the same training data (ProbFuse and SegFuse) and one that does not rely on training data (CombMNZ).

This was achieved by using a sliding window to use the probable relevance of a document's neighbours to estimate the probability that a document itself is relevant.

At present, SlideFuse assumes that each result set returned by an input system is of the same quality, as the probabilities used for fusion will be same in each case. In the future, we aim to investigate methods of weighting a particular result set according to its quality. This could possibly involve the use of the scores assigned to each document as a measure of an input system's confidence in its own results. Another approach to weighting would be to introduce weights within the sliding windows themselves, so as to place more emphasis on those documents that are closest to the rank around which the window is centred. Finally, a minor drawback of SlideFuse is that documents returned in positions beyond the length of the training sets will not be taken into account when fusing. We aim to address this situation in a more satisfactory fashion.

References

1. Bartell, B.T., Cottrell, G.W., Belew, R.K.: Automatic combination of multiple ranked retrieval systems. In: SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, Springer-Verlag New York, Inc. (1994) 173–181 Reference to show that it has long been demonstrated that fusion improves results.
2. Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D., Frieder, O., Goharian, N.: Fusion of effective retrieval strategies in the same information retrieval system. J. Am. Soc. Inf. Sci. Technol. **55**(10) (2004) 859–868
3. Vogt, C.C., Cottrell, G.W.: Fusion via a linear combination of scores. Information Retrieval **1**(3) (1999) 151–173
4. Aslam, J.A., Montague, M.: Bayes optimal metasearch: a probabilistic model for combining the results of multiple retrieval systems. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2000) 379–381
5. Voorhees, E.M., Gupta, N.K., Johnson-Laird, B.: The collection fusion problem. In: Proceedings of the Third Text REtrieval Conference (TREC-3). (1994) 95–104
6. Lillis, D., Toolan, F., Collier, R., Dunnion, J.: ProbFuse: a probabilistic approach to data fusion. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, New York, USA, ACM Press (2006) 139–146
7. Shokouhi, M.: Segmentation of search engine results for effective data-fusion. In: Proceedings of the 29th European Conference on Information Retrieval Research (ECIR 2007), Rome (April 2-7 2006)
8. Fox, E.A., Shaw, J.A.: Combination of multiple searches. In: Proceedings of the 2nd Text REtrieval Conference (TREC-2), National Institute of Standards and Technology Special Publication 500-215. (1994) 243–252
9. Callan, J.P., Lu, Z., Croft, W.B.: Searching distributed collections with inference networks. In: SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (1995) 21–28
10. Si, L., Callan, J.: Using sampled data and regression to merge search engine results. In: SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2002) 19–26

11. Montague, M., Aslam, J.A.: Condorcet fusion for improved retrieval. In: CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management, New York, NY, USA, ACM Press (2002) 538–548
12. Lee, J.H.: Analyses of multiple evidence combination. SIGIR Forum **31**(SI) (1997) 267–276
13. Voorhees, E.M., Gupta, N.K., Johnson-Laird, B.: Learning collection fusion strategies. In: SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (1995) 172–179
14. Aslam, J.A., Montague, M.: Models for metasearch. In: SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, ACM Press (2001) 276–284
15. Craswell, N., Hawking, D., Thistlewaite, P.B.: Merging results from isolated search engines. In: Australasian Database Conference, Auckland, New Zealand (1999) 189–200
16. Lawrence, S., Giles, C.L.: Inquirus, the NECI meta search engine. In: Seventh International World Wide Web Conference, Brisbane, Australia, Elsevier Science (1998) 95–105
17. Gravano, L., Chang, K., Garcia-Molina, H., Paepcke, A.: Starts: Stanford protocol proposal for internet retrieval and search. Technical report, Stanford, CA, USA (1997)
18. Lillis, D., Toolan, F., Collier, R., Dunnion, J.: Probabilistic data fusion on a large document collection. In: Proceedings of the 17th Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2006), Belfast, Northern Ireland, Queen's University Belfast (2006)
19. Craswell, N., Hawking, D.: Overview of the TREC-2004 web track. In: Proceedings of the Thirteenth Text REtrieval Conference (TREC-2004). (2004)
20. Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2004) 25–32
21. Silverstein, C., Henzinger, M., Marais, H., Moricz, M.: Analysis of a Very Large AltaVista Query Log. Technical Report 1998-014, Digital SRC (1998) http://gatekeeper.dec.com/pub/DEC/SRC/technical-notes/abstracts/src-tn-1998-014.html.