UCD SIFT in the TREC 2009 Web Track

David Lillis[1], Fergus Toolan[1], Ling Lin[2], Rem W. Collier[1], and John Dunnion[1]

[1] School of Computer Science and Informatics
University College Dublin
Ireland
{david.lillis, fergus.toolan, rem.collier, john.dunnion}@ucd.ie
[2] The Software School
Fudan University
Shanghai
China

Abstract. The SIFT (SIFT Information Fusion Techniques) group in UCD is dedicated to researching Data Fusion in Information Retrieval. This area of research involves the merging of multiple sets of results into a single result set that is presented to the user. As a means of evaluating the effectiveness of this work, the group entered Category B of the TREC 2009 Web Track.
This paper discusses the strategies and experiments employed by the UCD SIFT group in entering the TREC Web Track 2009. This involved the use of freely-available Information Retrieval tools to provide inputs to the data fusion process, with the aim of contrasting with more sophisticated systems.

1 Introduction

This is the first year of the SIFT (SIFT Information Fusion Techniques) Project's participation in the TREC Web Track. As the first entry, it was decided to enter Category B, as the current computing resources of the group would have made participation in Category A difficult. The principal aim of the SIFT group is to develop data fusion algorithms that combine the outputs of multiple Information Retrieval systems or algorithms in order to produce a single result-set that is of a superior quality.

The approach taken to the TREC Web Track was to focus on a number of these fusion algorithms that have been developed within the group, three of which were used for the three runs that were submitted. This means that, in contrast to a typical TREC Web Track entry, the focus was not on the evaluation of novel IR systems or algorithms, but on techniques to combine these.

The paper is organised as follows: Section 2 briefly discusses the area of Data Fusion. The specific algorithms used in the TREC experiments are discussed in greater detail in Sections 3, 4 and 5. The experiments themselves are described in 6. Finally some evaluation is contained in Section 7, in addition to outlining further experiments that will provide a more informative evaluation.

2 Data Fusion

Data Fusion refers to a procedure whereby a variety of result sets produced by a number of distinct Information Retrieval systems are merged into a single result that is then presented to the user [1]. It specifically refers to a situation where these input systems retrieve documents from the same document collection, so that an identical set of documents is available for retrieval by each.

A typical data fusion algorithm will attempt to exploit one or more of the "effects" identified in [2]. The "Chorus Effect" argues that documents retrieved by multiple input systems are more likely to be relevant and so a fusion algorithm should boost documents on whose relevance the systems agree. The "Skimming Effect" is based on the observation that relevant documents are most frequently to be found in early positions in result sets, and so by "skimming" the top documents from each result set (or attaching greater weight to these), a fusion technique should boost its performance. Finally, the "Dark Horse Effect" occurs when an input system produces unusually accurate or unusually inaccurate results. This latter effect is appealed to far less frequently than the others, due to the inherently difficult nature of identifying when such an event has occurred.

Traditionally, approaches to data fusion have tended to fall broadly into two categories. Rank-based algorithms make use of the position a document occupies in a result set in calculating the final ranking. Such algorithms include approaches based on interleaving [3] and voting-based techniques [4, 5]. The other major category is algorithms that makes use of the score used by the input systems to rank the documents. This includes linear combination models [2, 6] and the widely-used CombSum and CombMNZ algorithms [7, 8].

More recently, some researchers have focussed on making use of probabilities to rank fused result sets. Such approaches include Bayes-fuse [1], ProbFuse [9, 10], SegFuse [11] and SlideFuse [12]. Two of these algorithms, namely ProbFuse and SlideFuse, are the techniques used in the experiments described in this paper. They are discussed in more detail in Sections 3 and 4 respectively.

A more thorough discussion of existing Data Fusion techniques can be found in [9].

3 ProbFuse

ProbFuse is a probabilistic data fusion algorithm that makes use of training data to estimate the probability that a document returned in a particular segment of a result set is relevant. It is discussed in detail in [9, 10], where experiments on various TREC datasets demonstrate superior performance when compared with the popular CombMNZ data fusion algorithm [7].

The ProbFuse algorithm consists of two distinct phases. Firstly, a training phase is required so as to construct a model of the probabilities that particular documents are relevant. Once this has been done, a Fusion Phase performs the actual data fusion, thereby generating the final result sets to be returned to the user. These two phases are discussed in the following sections.

3.1 Training Phase

Because ProbFuse depends on probability to rank documents in the fused result set, it is necessary to estimate the probability of relevance for particular documents. In this algorithm, this is done by examining the result sets generated in response to a number of training topics. This approach relies on the assumption that the performance of a particular input system on such training topics is representative of its quality when faced with future topics.

ProbFuse requires that training queries be run on each system that provides result sets for fusion. By doing this, a separate set of probabilities can be built up for each. Thus, those systems that tended to produce superior results in the training phase will be boosted above the others in the fusion phase.

The model of probability used by ProbFuse requires that each result set be divided into x *segments* of equal size, where possible. In some cases when the result set is not evenly divisible by x, some rounding is necessary. The division of a simple 12-document result set into segments is illustrated in Figure 1. Here, the leftmost result set is seen to be divided into two segments, with half of the documents appearing in each. Subsequent examples show increasing values for x, resulting in greater numbers of segments being created.

The object of the training phase of ProbFuse is to ascribe probabilities to each segment that will represent the likelihood that a document appearing in that segment will be relevant to any given topic. As an example, in the case where $x = 2$, the probability of a document being relevant when returned by the system in question will depend on whether it appears in the first or the second half of the returned result set. The division of result sets into segments, rather than depending on actual rankings, allows for variations in the lengths of the result sets being used.

Mathematically, the training phase of ProbFuse is represented as follows: in a training set of T topics, $P(d_k|m)$, the probability that a document d returned in segment k is relevant, given that it has been returned by retrieval model m, is given by:

$$P(d_k|m) = \frac{\sum_{t=1}^{T} \frac{R_{k,t}}{K}}{T} \tag{1}$$

where $R_{k,t}$ is the number of documents in segment k that are judged to be relevant to topic t, and K is the total number of documents in segment k.

For each input system, it is necessary to calculate a probability of relevance for every segment k in the range $1..x$.

3.2 Fusion Phase

Once a set of probabilities has been calculated for each system being used for fusion, the fusion phase may take place. In this phase, each document is examined and its position in each of the result sets to be fused is noted. Depending on the segment the document is returned in, each system may then contribute towards

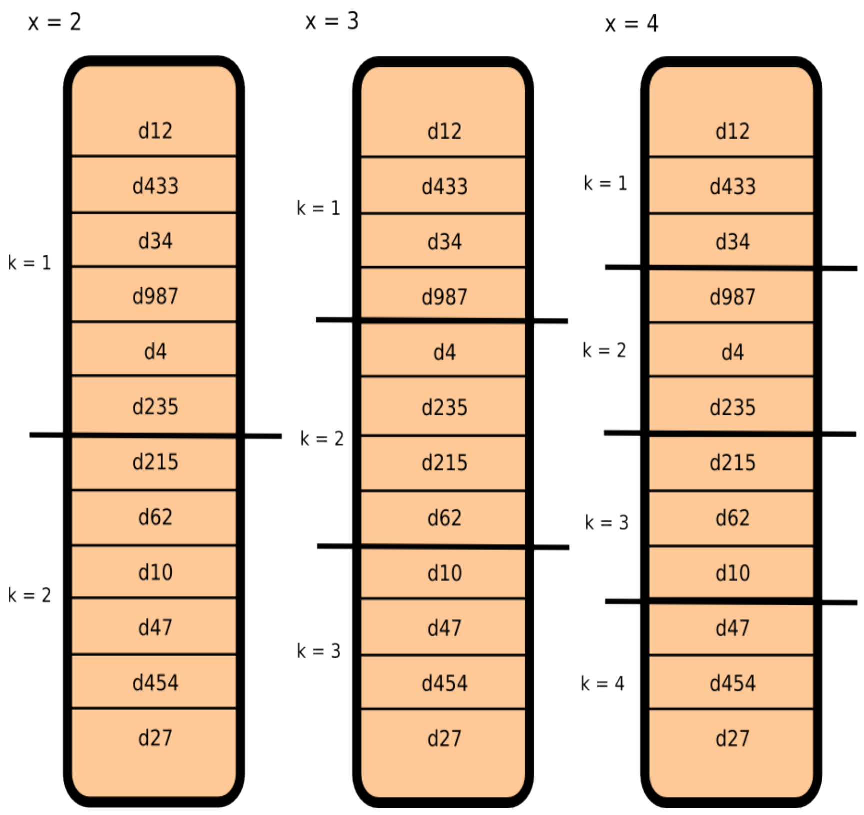

Fig. 1. Segmenting a result set for different values of x

that document's final ranking score, with no contribution occurring from any system that fails to return the document.

The ranking score S_d for each document d is given by

$$S_d = \sum_{m=1}^{M} \frac{P(d_k|m)}{k} \tag{2}$$

where M is the number of retrieval models being used, $P(d_k|m)$ is the probability of relevance for a document d_k that has been returned in segment k in retrieval model m, and k is the segment that d appears in (1 for the first segment, 2 for the second, etc.). For any technique that does not return document d in its result set at all, $P(d_k|m) = 0$, so as to ensure that documents do not receive any boost to their ranking scores from techniques that do not return them as being judged relevant.

Once S_d has been calculated for each document, the documents are then merged into the final result set, sorted in descending order of S_d.

4 SlideFuse

SlideFuse is an alternative probabilistic fusion algorithm that operates on a similar premise to ProbFuse. It also calculates ranking scores based on the probability that a document is relevant, where this probability is estimated by examining past performance in response to training topics. However, this probability distribution is calculated in a different way to that of ProbFuse. In previous work, SlideFuse has been shown to achieve superior performance to ProbFuse on the TREC2004 Web Track, when measured using both the MAP and bpref evaluation metrics [12].

As with ProbFuse, SlideFuse consists of two distinct phases: a Training Phase and a Fusion Phase. These are discussed in the following sections.

4.1 Training Phase

Unlike the training phase used in ProbFuse, SlideFuse does not divide result sets into segments. Instead, it attempts to approximate the probability of relevance at each position in a result set. However, this is not a trivial task for a number of reasons. Principal amongst these is the problem of incomplete relevance judgments amongst data sets that could realistically be used for training purposes. This can result in a situation where the probability of relevance in a particular position may appear to be zero as a result of failing to return any judged relevant documents in that position. This however can result in a jagged probability distribution, as illustrated in Figure 2. For this reason, an extra stage is required in the fusion phase that smooths these probabilities into a more realistic probability distribution.

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

$$P(d_p|s) = \frac{\sum_{t \in T_p} R_{d_p,t}}{T_p} \tag{3}$$

where T_p is the set of all training topics for which at least p documents were returned by the input system and $R_{d_p,t}$ is the relevance of the document d_p to topic t (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

As noted above, using the probability at each position tends to lead to jagged probability distributions, due principally to the presence of unjudged documents in each result set that are 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

$$a = \begin{cases} p - w & p - w >= 0 \\ 0 & p - w < 0 \end{cases} \tag{4}$$

$$b = \begin{cases} p + w & p + w < N \\ N - 1 & p + w >= N \end{cases} \tag{5}$$

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.

Having defined the limits of the window surrounding each position in the result set, it is necessary to assign a probability score to each position based on the contents of its window. This is done as follows: $P(d_{p,w}|s)$, 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

$$P(d_{p,w}|s) = \frac{\sum_{i=a}^{b} P(d_i|s)}{b - a + 1} \tag{6}$$

The use of the window results in the smoother probability distribution shown in Figure 2. It also diminishes the likelihood of a rank being assigned a probability score of zero, even where relevant documents have occurred in surrounding areas.

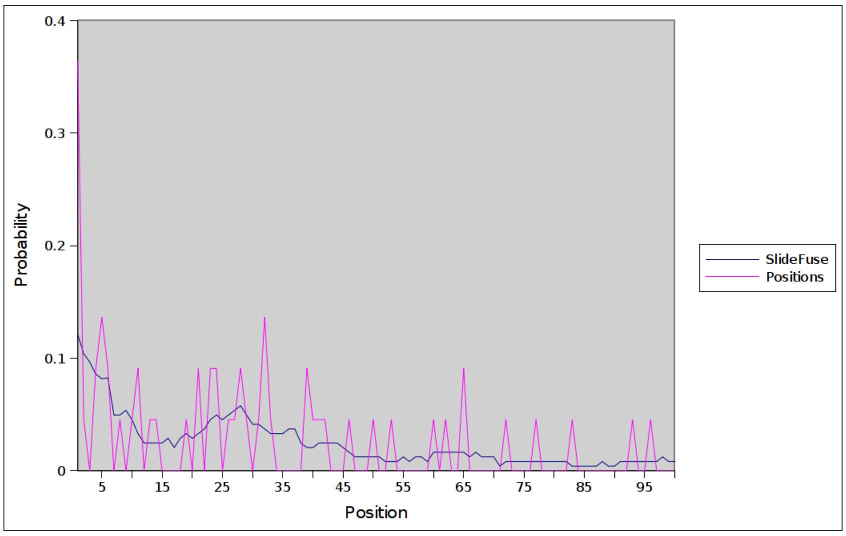

Fig. 2. Probability Distribution using SlideFuse

The final stage of the fusion phase is to assign a ranking score to each document that will be used to order the fused result set at the end of the process. R_d, the final ranking score given to document d is given by

$$R_d = \sum_{s \in S} P(d_{p,w}|s) \tag{7}$$

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.

5 Interpolated SlideFuse

Interpolated SlideFuse is a slight variation on the regular SlideFuse algorithm outlined above. Despite the smoothing of the achieved by the introduction of the sliding window surrounding each rank, the resultant probability distributions do not always reflect the general observation of the Skimming Effect that relevant documents are more likely to occur in early positions in a result set. Figure 2 is an example of such a situation, where the probability for the windows surrounding ranks 25-35 tend to be higher than those in the earlier 10-20 positions.

For this reason, an additional interpolation step is added to the existing SlideFuse algorithm when the probability associated with each window is calculated. Rather than merely considering the probabilities associated with ranks in the immediate surrounding ranks, care is also taken so as to ensure a downward-sloping curve. The probability assigned to each window is the maximum probability of the window itself and all other windows occurring further down the result set. Thus, for a result set of 1000 documents, the probability associated with window number 100 is the maximum probability found between window 100 and window 1000, inclusive. An in-depth study of the effects of this change has not yet been carried out, and its use in the TREC 2009 Web Track is its first proper use in an experimental context.

6 TREC 2009 Experiments

On account of the resources available to the group, it was decided to enter Category B, which consists of an adhoc task and a diversity task. The dataset used is a subset of English-only pages from the ClueWeb09 dataset, containing approximately 50,000,000 web pages.

In designing the experiments for the Web Track, a number of significant issues arose and had to be addressed. In particular, these were:

– **Inputs**: All data fusion algorithms require at least two result sets in order to operate. In order to utilise the data fusion techniques outlined above, it was necessary to generate numerous result sets for each topic.
– **Training Data**: One limitation of probabilistic data fusion algorithms is the need for training data. This meant that in addition to using several systems to generate results, it was also necessary to have a historical record of their effectiveness for training purposes.

6.1 Inputs

The Terrier (TERabyte RetrIEveR) is an open-source search engine developed in the University of Glasgow and released under the Mozilla Public License [13]. Terrier is specifically designed to be capable of handling large-scale document collections, in the order of terabytes. This, allied with the fact that it offers implementations of a variety of document ranking models, made it an attractive choice for providing the inputs to the fusion process.

To identify which of the available models were to be used in the Web Track experiments, each was run on the WT10G dataset using TREC-10 topics 501-550 and the MAP scores were compared. As a result of this, the four models with the best MAP scores on these topics were selected to generate the inputs for the fusion experiment. There were BB2, IFB2, In_expB2 and InL2.

6.2 Training Phase

From the previous Sections, it can be seen that each of the data fusion algorithms used requires training data, so that the probability of relevance can be estimated. When training, it is essential that the systems running the training queries are the same as those used to ultimately generate the result sets for fusion.

For each of the chosen document weighting models, TREC-10 topics 501-550 (title only) were run on the WT10G dataset. This resulted in 50 training result sets being generated from each input system, with probabilities being calculated during a training phase for each algorithm, as outlined in the sections above.

For ProbFuse, it is necessary at this stage to specify x, the number of segments into which each result set is divided. Here, x was set to 25: a value that has produced satisfactory empirical results in past experiments [9].

6.3 Fusion Phase

In the fusion phase, each of the four selected models was used to generate a result set for each of the topics. In generating these result sets, only the standard stopword removal and stemming (using Porter's algorithm [14]) were used. Query Expansion, Relevance Feedback and other such techniques were not used.

In this phase, it was necessary to set the size of the window to be used in SlideFuse and Interpolated SlideFuse. For this experiment, this was set to 5, following previous experiments in [12].

7 Evaluation and Future Work

A summary of the evaluation results is contained in Tables 1 and 2. These indicate the comparative evaluation scores of the SlideFuse (UCDSIFTslide), ProbFuse (UCDSIFTprob) and Interpolated SlideFuse (UCDSIFTinter) when compared with the other participants. In total, Category B had 34 submissions.

Table 1 shows the percentile ranks for each of the algorithms used when evaluated using the MTC sampling methods [15]. Here, it can be seen that the performance of each algorithm does not achieve a high rank for overall evaluation metrics such as eMAP and Rprec. However, the ranks are much higher in each case for metrics that measure precision in early positions. Each algorithm achieves its highest rank for either P5 or P10, with deteriorating performance further down result sets.

Table 1. MTC measures: percentile rank (34 participants)

	eMAP	Rprec	P5	P10	P15	P20	P30	P100
UCDSIFTslide	8	8	76	79	67	58	47	8
UCDSIFTprob	11	11	41	50	47	32	20	14
UCDSIFTinter	5	5	82	76	64	55	41	11

Table 2 shows similar data for StatAP measures [16]. Here again, MAP and Rprec ranks are not high. Once more, however, the rankings for precision in early positions (P30 in this instance) are substantially higher. Despite the low overall rankings, the more competitive early precision results are encouraging given the tendency of users to only examine a small number of results at the top of a result set [17].

Table 2. StatAP measures: percentile rank (34 participants)

	MAP	Rprec	P30	NDCG
UCDSIFTslide	5	5	79	32
UCDSIFTprob	8	8	52	26
UCDSIFTinter	11	11	82	29

The principal area of interest for further evaluation is in the runs submitted by other groups for the TREC-2009 Web Track, in both Categories A and B. These represent sophisticated, state-of-the-art systems that employ a variety of approaches to the task and form a diverse range of inputs that can be used in a data fusion process. Also, since only standard document-ranking models were used in our initial experiments, the performance of the runs submitted by other groups is likely to be higher than for those generated by Terrier. We are especially interested in evaluating the effectiveness of our data fusion approach on these inputs and making comparisons with the runs we ourselves have submitted.

References

1. 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 (2000) 379–381

2. Vogt, C.C., Cottrell, G.W.: Fusion Via a Linear Combination of Scores. Information Retrieval **1** (1999) 151–173
3. 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
4. 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 (2001) 276–284
5. 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 (2002) 538–548
6. 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 (1995) 21–28
7. 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
8. Lee, J.H.: Analyses of multiple evidence combination. SIGIR Forum **31** (1997) 267–276
9. 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 (2006) 139–146
10. Lillis, D., Toolan, F., Collier, R., Dunnion, J.: Probabilistic Data Fusion on a Large Document Collection. Artificial Intelligence Review (2007)
11. Shokouhi, M.: Segmentation of Search Engine Results for Effective Data-Fusion. Advances in Information Retrieval **4425** (April 2007)
12. Lillis, D., Toolan, F., Collier, R., Dunnion, J.: Extending Probabilistic Data Fusion Using Sliding Windows. In: Proceedings of the 30th European Conference on Information Retrieval (ECIR '08). Volume 4956 of Lecture Notes in Computer Science., Berlin, Springer (2008) 358–369
13. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Johnson, D.: Terrier information retrieval platform. In: Proceedings of the 27th European Conference on Information Retrieval (ECIR 05). (2005) 517–519
14. Porter, M.F.: An algorithm for suffix stripping. (1997) 313–316
15. Carterette, B., Allan, J., Sitaraman, R.: Minimal test collections for retrieval evaluation. Annual ACM Conference on Research and Development in Information Retrieval (2006)
16. Aslam, J.A., Pavlu, V.: A practical sampling strategy for efficient retrieval evaluation
17. Silverstein, C., Henzinger, M., Marais, H., Moricz, M.: Analysis of a Very Large AltaVista Query Log. Technical report (1998)