

# Internet Multimedia Search and Mining

Xian-Sheng Hua, Marcel Worring, and Tat-Seng Chua

May 10, 2012



# Contents

<b>Contents</b>	<b>I</b>
<b>List of Figures</b>	<b>I</b>
<b>List of Tables</b>	<b>I</b>
<b>1 Weighted Data Fusion for CBMIR</b>	<b>1</b>
Introduction . . . . .	1
Relationship to Internet Media Search . . . . .	3
Terminology . . . . .	4
Data Fusion . . . . .	7
Experimental Setup . . . . .	8
Retrieval Experts . . . . .	10
Data Collections . . . . .	11
Experimental Methodology . . . . .	14
Exploration of Data Fusion Variables . . . . .	16
Optimal Distribution of Weights . . . . .	17
Combination Levels . . . . .	23
Equivalence Transformations . . . . .	28
Combination Operators . . . . .	32
Discussion . . . . .	34
Current Approaches for Weight Generation and Data Fusion . . . . .	35
Query Independent Weighting . . . . .	35
Query-Class Weighting . . . . .	36
Discriminative Classification . . . . .	36
Other Methods . . . . .	38
Relevance Feedback . . . . .	38
Query Performance Prediction . . . . .	39
Conclusions . . . . .	40
<b>References</b>	<b>43</b>



# List of Figures

1.1	Early and Late Fusion Approaches . . . . .	8
1.2	Average Expert Performance . . . . .	12
1.3	Experimental Models . . . . .	15
1.4	Histogram of weight distribution, all corpora . . . . .	18
1.5	Q-Q Plots of Weight Distributions . . . . .	20
1.6	Highly Weighted Pairs, all corpora . . . . .	22
1.7	Combination Levels . . . . .	24
1.8	Experimental Results, all corpora . . . . .	28
1.9	Query Class Weighting . . . . .	37



# List of Tables

1.1	Example System and Query . . . . .	6
1.2	Central Tendency Measures of Ideal Weights . . . . .	19
1.3	Distribution of $1\sigma$ Weights Amongst Experts . . . . .	21
1.4	Normalisation Experimentation, all corpora . . . . .	30
1.5	CombSUM and CombMNZ Evaluation . . . . .	33
1.6	CombSUM vs CombMNZ behaviour . . . . .	34
1.7	Summary of Major Findings . . . . .	41





## Chapter 1. Weighted Data Fusion for CBMIR

Peter Wilkins

CLARITY: Centre for Sensor Web Technologies, Dublin City University, Ireland

Alan F. Smeaton

CLARITY: Centre for Sensor Web Technologies, Dublin City University, Ireland

**Abstract:** In this chapter we present an overview of *data fusion*, and how it can be applied to the task of internet multimedia search, specifically content-based multimedia search. The chapter will primarily be focused on the *weighted* combination of ranked results from different retrieval experts, to formulate a final ranking for some given content-based information need. The types of data under examination in this chapter are *low-level* multimedia features, such as colour histograms, edge detection etc. The chapter reports an extensive series of experiments on a sizable collection of visual media and from these experiments a set of interesting and surprising results emerge.

### *Introduction*

The availability of information resources on the internet has ushered in the ‘Web 2.0’ phenomenon, spearheaded by websites which are ‘mashups’. These are websites which combine forms of data from multiple external sources in order to fulfill some form of information need. Often these forms of data will be multimedia, and allow for the creation of rich, informative sources of information. In many ways, the ‘mashup’ can be seen as an extension of an earlier web phenomenon, the meta-search engine which still exists today, as seen in a search service such as ‘dogpile.com’. The meta-search engine takes in a single information query and combines the outputs of multiple other external search services to formulate a single response to that query. Both of these tasks, mashup and meta-search, are executing an operation known as *Data Fusion*.

Data fusion is defined as “the combination of evidence from differing systems” [1] with the aim of maximizing retrieval performance. In this chapter, we focus on data fusion, specifically with reference to multimedia search applications, where the types of data to be combined are considered to be very noisy. The fundamental purpose of this chapter is to demonstrate that even with very poor performing sources of data which are combined, retrieval performance through data fusion can achieve far greater results than over using single resources alone. Furthermore, the data fusion task we consider in this chapter is the combination of results from multiple search engines, that is each source of evidence is a ranked list of results, each of which is generated from some search service for the same information need.

On initial inspection, the data fusion task may appear relatively straightforward, namely to accept a query, issue it to search service ‘a’ and search service ‘b’ and merge the results. However there are many factors which impact upon the quality of this combination. For instance, how many results from each search service should be taken ? Is there a difference in quality between the search services, such that one service should be given a greater weight than another ? If so, how do we determine this weight ? Are the ranking metrics of each of the search services equivalent, or will they require some normalization first ? Should the combination be a summation of the scores assigned to each document, or its rank, and should this be a linear operation ? Each of these steps can impart significant performance benefits or degradations to retrieval performance. This chapter highlights each of these factors, and demonstrate the performance implications each factor brings.

Specifically let us define the type of system we are exploring in this chapter as a Content-Based Multimedia Information Retrieval (CBMIR) service, which is leveraging multiple ‘low-level’ search services, otherwise known as retrieval experts (i.e. each low-level search service is now referred to as a retrieval expert, or expert). The system will allow for ‘query-by-example’ (QBE), specifically multi-example QBE. A QBE query is one which allows a user to provide an example of what they are trying to find. For instance, in an image search engine, a QBE query image is submitted to that search engine such that the user’s intent is to find images which are visually similar to the query image. We are also making the assumption in this chapter that each of the retrieval experts we use has indexed the same set of data. For instance if we have 1,000 images, we may have a color retrieval expert, and an edge histogram retrieval expert. Both of these experts provide a different interpretation of the source data, but both are operating over the same 1,000 images. Finally, in this chapter, our search service will operate over a *broad* domain, rather than a narrow domain [12]. This means that the system will be generalized and it makes no specific assumptions about the data it is operating on.

Other chapters in this book have used low-level features for tasks such as concept detection. In this chapter, we are using multiple low-level features directly for retrieval through weighted data fusion. There are several reasons for this. Firstly, the direct use of low-level features facilitates the QBE paradigm, which offers a unique vector for a user to search a collection. Secondly, traditional low-level features have *individually* provided very poor retrieval performance [12], however we will demonstrate that through ideal weighted combination with data fusion, that excellent performance is achievable. Finally, by using noisy, low-level features, they act as a proxy for a variety of information search services that a system-builder may wish to incorporate. As this chapter will not tailor the investigation towards any individual low-level expert, it means that a researcher or system-builder reading this book can take the approaches discussed and apply them to any form of ranked information resources which they may wish to combine.

### *Relationship to Internet Media Search*

Throughout this book the central theme has been the study and application of techniques to assist in internet media search, where we can imagine services which interface with some form of consumer media, such as Flickr. In this chapter we make a slight divergence from this area by operating within independent video databases which people may wish to search. That is, we can imagine that we have a silo of digital video (for instance, archive news video) that we wish to search within. Whilst this is the data which is used in this chapter, the techniques and analysis discussed here are of wider application and indeed can be applied to the more general category of internet media search. For instance, new search services are appearing which call themselves ‘reverse search engines’, such as [tineye.com](http://www.tineye.com)<sup>1</sup>, which are essentially content-based image search engines that allow for QBE. Other search services which may be developed could involve complex media searches, such as within blogs, where both the text and the images are analysed at a content-level, to allow for a richer vein of searching. In both of these instances we can see that the analysis of the data would involve several different types of content-analysis, and therefore for any retrieval task their successful combination. Such a combination is the focus of this chapter. When reading this chapter, the reader may consider each type of ‘expert’ to be atomic and as such the techniques discussed in this chapter can be considered broadly generic. The conclusions reached within this chapter however may have some specificity to the data which was used, however they will inform the reader of potential pitfalls to avoid, or at the very least investigate when constructing their system. Indeed the general study of ‘data fusion’ is widely applicable to many research domains, and the techniques used in data fusion can be considered as being independent of any particular type of data.

Within this book, the techniques described within the chapter titled “Cross-modality Indexing, Browsing and Search of Distance Learning Media on the Web (Arnon Amir, Kobus Barnard, and Alon Efrat)” describe alternative approaches to tackling the content-based search problems described within the present chapter.

The chapter is organized as follows. Following from this section, we will briefly examine data fusion itself providing some context and history for its use. Next in Section 1 we will provide an overview of the experimental framework used to generate our observations, and discuss the datasets, system and its resources used in this investigation. Section 1 will present our examination of data fusion and its variables, describing the variables which can be modified and their effect on performance. After this, Section 1 will present current approaches which use data fusion. Finally in Section 1 we will briefly examine related areas of research which can benefit from the observations presented in this chapter, followed by our conclusions.

---

<sup>1</sup><http://www.tineye.com>

### **Terminology**

For reasons of clarity we will now formally define the terms used throughout the remainder of this chapter. We assume that a CBMIR system combines multiple retrieval experts together for queries which contain multiple components. For these definitions we use set and matrix notations, however our use of matrix notation is slightly unorthodox, as matrices typically contain numbers. In our notation, the matrices we define will contain ranked lists of documents. The use of matrix notation is for conceptual reasons to assist the reader in comprehending the number of variables at work.

**Retrieval Expert** A Retrieval Expert is treated as a black box, it is a service which can process a query and return a ranked set of documents on a given collection. Formally our system will have Retrieval Experts  $E = \{expert_1 \dots expert_i\}$  where  $1 \leq i \leq |E|$ .

**Query** A Query within our context is a multi-modal request which describes an information need and is processed by the CBMIR system. A query may be comprised of text, multiple visual examples, etc. Within this context, each component of a query is treated as a separate query to be processed, e.g. a text component would go to the text expert, each individual visual component would go to visual experts. Therefore a Query  $Q$  is comprised of  $\{query_1 \dots query_j\}$  where  $1 \leq j \leq |Q|$ . As an example, if a user is searching for pictures of boats, the query may be the text ‘boats’ and two images of boats, therefore in this case the individual components of the query are  $\{‘boats’, image_a, image_b\}$  otherwise referenced as  $\{query_1, query_2, query_3\}$ , each of which may be individually processed by the retrieval experts available. A multi-QBE query allows for a degree of query disambiguation. For instance, if a user wants to find images of a flower, but does not care about color, s/he can issue a query for a red flower and a yellow flower, indicating that they want a flower surrounded by green, but that the color of the flower itself is not as important [20].

**Documents** Documents in this context are the semantic unit of information that is indexed and retrieved. Typically the term refers to entire text documents or web-pages for ranking. In the case of MIR, the unit of retrieval can be a video, a ‘shot’ (i.e. a segment of video which is visually consistent), an audio recording, etc. The use of the term ‘documents’ will refer to any retrieval unit that can be handled by a CBMIR system.

**Result Set** A result set is the product of a unique pair of Retrieval Expert and Query  $\langle expert_i, query_j \rangle$ , which produces an ordered set of documents  $R$  such that  $R = \{document_1 \dots document_m\}$  where  $1 \leq m \leq |R|$ . Every unique pair  $\langle expert_i, query_j \rangle$  produces a Result Set, which collectively form the matrix  $\mathbf{RS} = [rs_{i,j}]$   $i = 1 \dots |E|, j = 1 \dots j = |Q|$ . The row index  $i$  represents experts, while column index  $j$  represents query components. Therefore,  $rs_{i,j}$  represents the result set  $R$  generated by expert  $i$  and query component  $j$ .

Every  $document_m$  in  $R$  will have associated with it a triple  $\langle name, rank, score \rangle$  where  $name$  is a unique identifier for the document,  $rank$  is in the set  $\mathbb{N}$  and  $score$  is in  $\mathbb{R}$ . Every value of  $rank$  will be unique in the set and it is desirable that every  $score$  value is also unique, however depending on the ranking function this may not always hold. As the set  $R$  is ordered, the values of both  $rank$  and  $score$  will change monotonically as one iterates through the set. In order to compute a final response to a query, a set of coefficients is typically required so that we can give greater weight to those sets  $R$  which are likely to enhance retrieval performance.

**Retrieval Coefficients** We define a matrix of weights which are used to alter the impact of different  $rs_{i,j}$  when they are combined into a single result. This matrix is  $\mathbf{RC}$ , where  $\mathbf{RC} = [rc_{i,j}] i = 1 \dots |E|, j = 1 \dots j = |Q|$ . Individual coefficients have the properties  $rc_{i,j} \in \mathbb{R}^+$  and  $\sum rc_{i,j} = 1$ . Every result set in matrix  $\mathbf{RS}$  will have a corresponding entry in  $\mathbf{RC}$ . For instance if no weighting was desired, all entries in  $\mathbf{RC}$  would be set to the same value, thus providing a uniform weighting.

The final result set therefore, is some combination of the result sets  $rs_{ij}$  generated by pairs  $\langle expert_i, query_j \rangle$  from sets  $E$  and  $Q$ , and application of the retrieval coefficients  $\mathbf{RC}$ . Taking our previous example of finding images of boats with an CBMIR system which has available 3 retrieval experts (one text expert, two visual), we have potentially 5 result sets to combine into a single result set (one text result and four visual results), and up to 5 weights that can be applied.

To help illustrate the various approaches that different weighting schemes employ and how they impact upon retrieval, we will refer back to this section to help illustrate how various approaches may work. Defined in this section is an example CBMIR system and a multi-example query to it, as shown in Table 1.1.

In this example system, there are four retrieval experts ( $E$ ) available within the system (where each of these experts is considered a black box with its own index and ranking function). We have also defined a query ( $Q$ ) that is issued to the CBMIR system, which consists of the text “flowers”, an image of a red flower and an image of a yellow flower. The system presented with this query, generates seven result sets ( $R$ ), one for the text query, then six more from the two query images against the three visual experts. Technically there are more instances  $R$  than listed here, such as querying a visual expert with the text query, however this will produce a null set of results which for reasons of clarity we do not show here.

To summarise, we have defined within our CBMIR system, retrieval experts  $E$ , multi-part queries  $Q$ , individual result sets  $R$  and the matrix which contains all result sets  $\mathbf{RS}$ . To weight each ranked list contained in the matrix  $\mathbf{RS}$  we have the retrieval coefficients matrix  $\mathbf{RC}$  which is used to weight each ranked lists so that they can be combined into









Example System and Multi-Example Query			
$E$	Expert Set	$Q$	Query Set
$expert_1$	Text Expert	$query_1$	"Flowers"
$expert_2$	Colour Expert	$query_2$	
$expert_3$	Edge Expert	$query_3$	
$expert_4$	Texture Expert		
$ E $	4	$ Q $	3
Result Set matrix <b>RS</b>			
$rs_{1,1}$	"Flower" $\mapsto$ Text Expert	$rs_{3,2}$	 $\mapsto$ Edge Expert
$rs_{2,2}$	 $\mapsto$ Colour Expert	$rs_{3,3}$	 $\mapsto$ Edge Expert
$rs_{2,3}$	 $\mapsto$ Colour Expert	$rs_{4,2}$	 $\mapsto$ Texture Expert
		$rs_{4,3}$	 $\mapsto$ Texture Expert
Non-zero entries in matrix <b>RS</b> : 7			
Retrieval Coefficients matrix: <b>RC</b>			
rows $i$ experts, columns $j$ query components			
$\begin{pmatrix} rc_{1,1} & 0 & 0 \\ 0 & rc_{2,2} & rc_{2,3} \\ 0 & rc_{3,2} & rc_{3,3} \\ 0 & rc_{4,2} & rc_{4,3} \end{pmatrix}$			
Non-zero entries in matrix <b>RC</b> : 7			

Table 1.1: Example system and query, where there are 4 experts in the system and the query has 3 components.

our final response to a query.

$$\begin{aligned}
E &= \{expert_1 \dots expert_i\} \\
Q &= \{query_1 \dots query_j\} \\
R &= \{document_1 \dots document_m\} \\
document_m &\mapsto (name, rank, score) \\
\mathbf{RS} &= [rs_{i,j}]_{|E| \times |Q|} \\
\mathbf{RC} &= [rc_{i,j}]_{|E| \times |Q|}
\end{aligned}$$

## Data Fusion

When we refer to *Data Fusion* we are conducting “the combination of evidence from differing systems” [1] with the aim of maximizing retrieval performance. This is a distinct from the *Collection Fusion* problem, which Voorhees *et al.* defines as the combination of “retrieval runs on separate, autonomous document collections that must be merged to produce a single, effective result” [2]. Fusion is an overloaded term, within the multimedia processing community it can also refer to *Information Fusion* which includes activities such as the combination of various modalities such as the information from a visible light camera and an Infra-Red camera into a single signal [3].

Data fusion is the combination of multiple rank lists of documents returned from various retrieval experts, such that a single ranking is produced. Research into data fusion has been on-going since 1979, with observations about the overlaps in documents returned from different ranking models [4, 5, 6]. In particular, Saracevic & Kantor found that as result sets were being combined, that the probability of a document being judged as relevant increased monotonically as it appeared in more result sets. This led in turn to the hypothesis by Lee as to why data fusion works: “different runs might retrieve similar sets of relevant documents but retrieve different sets of nonrelevant documents” [7]. In other words, when we combine ranked lists of documents, there should be some degree of overlap in the relevant documents, but far less overlap in the non-relevant documents. Therefore when the scores or ranks of the relevant documents are combined, they are promoted higher in a final ranking. We can observe early implementations of this in the Garlic system developed by IBM which combined multiple information systems for retrieval through fuzzy sets [8]. For the interested reader, further texts on the data fusion hypothesis include McCabe *et al.* [9] and Beitzel *et al.* [10].

Within the domain of multimedia search there are two approaches within data fusion for the combination of data known as early fusion or late fusion [11], these are conceptually illustrated in Figure 1.1. Early fusion is essentially the combination of data prior to indexing, meaning that data is first somehow aggregated, and then a ranking model is placed over this aggregated data. There are multiple examples of this type of system, such



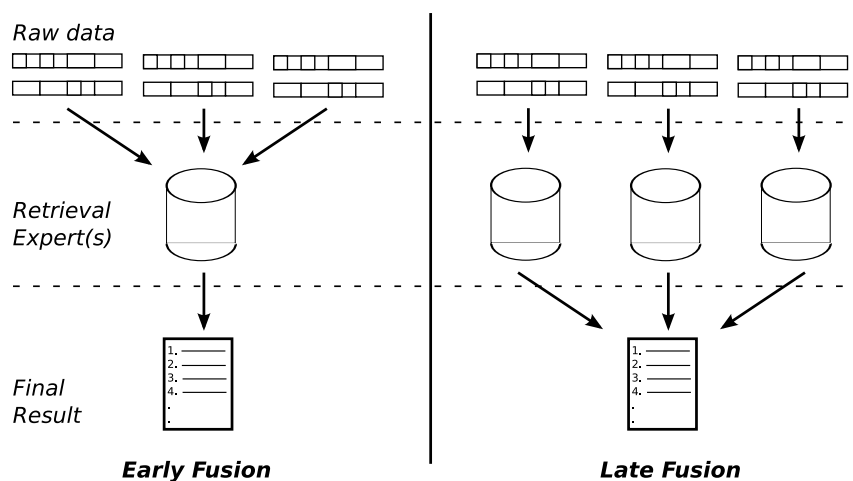


Figure 1.1: Early vs. Late Fusion

as combined document representations, classification tasks or learning to rank applications (some of which are discussed previously in this book). Therefore the combination of evidence through early fusion is typically handled in an ‘indexing’ phase. Late fusion instead assumes each source of data has associated with it some form of a ranking function, each of which can be independently queried. Once each source has been queried, the outputs of each of these queries can be aggregated together to form a final response to the initial query. Examples of late fusion include metasearch, and a majority of the text information retrieval experiments of combining ranked lists from different retrieval systems. Our choice of employing late fusion for our data fusion experiments means we do not need to explicitly model any particular source of data as that is the function of the retrieval expert associated with each source of data. This allows us to add or remove retrieval experts at any stage in the retrieval process, allowing for a relatively free environment in which to conduct our data fusion experiments. As such this is a generic approach which therefore has wide areas of application.

### ***Experimental Setup***

In this section we discuss our experimental framework for conducting our investigation into data fusion for multimedia retrieval. We begin first with a discussion of the retrieval experts which we use, followed by an overview of the data on which we conduct our experiments. Finally we discuss our experimental framework, which whilst unorthodox we believe is appropriate as it allows us to accurately observe the conditions under which weighted data fusion is successful for multimedia retrieval. First however we briefly describe the



evaluation metrics used in this chapter, more details of which can be found in [13, 14].

In summary, the two fundamental concepts that underpin most metrics used in the evaluation of search problems are *precision* and *recall*. Let us assume we have built a retrieval system, and it processes an incoming query. For that query we return a subset of documents to answer that query. In such a scenario we have the entire set of documents which the system knows about, which we call the ‘Collection’  $C$ . We have the subset of documents which were returned to the user, which we call  $n$ . We have as part of the collection the complete set of ‘correct’ (i.e. relevant) documents which answer the query, which we call  $R$ , and finally we have the subset of relevant documents which were returned to the user as part of  $n$ , which we refer to as  $r$ .

*Precision* is defined as the proportion of relevant documents contained in the set of documents which were returned to the user. We can express this as ‘the number of relevant documents returned’ divided by ‘the total number of documents returned’, or using the notation just defined:

$$Precision = \frac{r}{n} \quad (1.1)$$

So if we returned five documents to a user, and each of those five documents is relevant, then our value for precision would be 1. However, it may be for that query that there were in fact twenty documents which were of relevance to the user, but we only returned five. Precision did not capture this fact, that is the job of *recall*.

*Recall* is defined as the proportion of relevant documents returned to the user, out of all relevant documents contained in the collection. Using the notation we just defined:

$$Recall = \frac{r}{R} \quad (1.2)$$

Using the previous example, if we returned five relevant documents to the user out of a set of twenty relevant documents, then our recall value would be 0.25.

Both precision and recall can be considered as *set* based measures. That is there is no consideration given to the rank order that documents are returned in. Clearly for search problems if we are comparing two different search outputs, we would consider as better the search output which ranks its relevant documents higher in the ranked list.

The main evaluation measure we use is Average Precision (AP) which is designed with a bias of providing a better score for retrieval runs which rank relevant documents higher in a ranked list, over those which place relevant documents in lower rank positions, and is defined in Equation 1.3.

$$AveragePrecision = \frac{\sum_{n=1}^N Precision(n) \cdot Relevance(n)}{|Relevant|} \quad (1.3)$$

In AP the function  $Precision(n)$  is the precision value at  $n$  documents and  $Relevance(n)$  is a binary function which indicates if document  $n$  is relevant, variable  $N$  is the size of the ranked list and  $|Relevant|$  is the *total* number of relevant documents in the *collection*

under consideration. AP is the most common metric used in the evaluation of IR systems, however it provides a per *topic* score. Mean Average Precision (MAP) is a single scored metric which provides an indication of performance over an entire retrieval run, where a retrieval run consists of the results of multiple topics. MAP is simply the mean of all the AP scores in a retrieval run.

Alternative measures to AP exist, each of which has various advantages and disadvantages. A complete discussion of the evaluation metrics which could be used to evaluate search systems is out of the scope of this chapter, however the chapter “Evaluation of Multimedia Retrieval Systems” in Blanken *et al.* [14] provides a good overview.

Fundamental to all of these measures however, is the concept of a ‘ground truth’ otherwise known as ‘relevance judgements’. A relevance judgement is a pairing of a query to a list of documents that are considered as relevant to that query. For instance, we may want to evaluate our system on how well it can find cats. That means we need to define the query ‘cat’ and manually label our search collection for positive instances of cats. Once done, we can then run repeated experiments where we can measure the performance of our system in finding ‘cats’. The creation of relevance judgements is a research area within itself, for instance if you are searching over the entire web, it is clearly impossible to manually annotate every web page to determine if it is relevant to a query or not. However, that is the advantage of using ‘test collections’ such as we have used in this chapter for the purposes of system building and evaluation (see 1).

### ***Retrieval Experts***

The extraction of low-level features from multimedia data is analogous to the extraction of terms from text documents and can be considered as part of an indexing phase in a CBMIR system. Low-level feature extraction is typically a fully automatic process. We use the phrase ‘low-level’ to describe these features as they impart no semantic or higher level understanding of the data, but rather they output either data patterns or statistics about the data being analysed. Low-level features are the foundation of most CBMIR systems upon which either more advanced features can be built or retrieval systems constructed from. Their independent use for retrieval however, particularly ad-hoc retrieval in unconstrained domains illicit poor performance [12]. In general there are three major types of multimedia data, images, audio and video. Images are a static form of data, whilst both audio and video have a temporal component.

In this chapter, we make use of both visual experts, and text experts whose data has been obtained either through transformations of an audio signal of some video data (through Automatic Speech Recognition (ASR)), or from attached annotations to some visual data. The majority of the visual experts we employ in this work are derived from the MPEG7 XM (eXperimentation Model), a reference implementation of features described in the MPEG7 standard [21]. The visual features we used include Color Layout, Color Moments, Color Structure, Scalable Color, Edge Histogram and Homogeneous Texture

descriptors. Each of these also has an associated ranking metric which allows for the querying and generation of ranked lists by each of these features. Complete descriptions of these visual features can be found in Manjunath *et al.* [15]. For our text data we used a vector space model over this data to provide for retrieval functions, more details of which can be found in van Rijsbergen [13].

For illustrative purposes we conduct a simple experiment here where we measure the performance of each individual expert against each of our test corpora (discussed in the following subsection). The specifics of this test are that for each expert if there are multiple query components (e.g. the query contains multiple visual examples), then the results of these are uniformly weighted and combined to provide an average indication of that expert's performance. The results of this experiment are presented in Figure 1.2, with the x-axis representing the different test corpora which we will be using, and the y-axis representing Mean Average Precision (MAP). These results were generated by using MinMax normalization and CombSUM for combining multiple query-components within the one expert (i.e. as our queries are typically made up of three or more images, the result for a single expert is the aggregation of the results of those three queries for the one expert, the operators introduced here will be discussed in detail in the next section).

From this graph we can observe that individual retrieval experts perform very badly in terms of MAP across the majority of evaluation corpora, echoing previous research [12]. This is demonstrated by the majority of the results for each expert residing in the range 0.001 - 0.03 of MAP, which are very poor scores. Furthermore, we also note the performance variations which exist between experts, so whilst we have four color experts, in different retrieval scenarios, different experts perform better, indicating that it is beneficial to have multiple experts on hand and to combine their outputs.

We also note that there is significant performance shifts across corpora with different experts, demonstrating our motivation for experimenting on multiple corpora so as to gain more generalised knowledge of the behaviour of data fusion across a wide range of retrieval environments. This variation in performance though highlights the need for effective data fusion strategies, including the generation of appropriate weights for combination.

### **Data Collections**

For our experiments we are using what are known as 'test collections'. Test collections are standardized aggregations of data, queries and relevance judgements that allow for a uniform experimental environment. Basically, a standardized test collection allows us to more directly compare different systems, because it removes the impact that the data or the queries may have on performance. For instance, I may report that my system achieves excellent performance, but if my data collection consisted only of images of dogs, and my queries were about dogs, then it is not unexpected that I would achieve good performance. Standardized test collections provide a way in which multiple systems can be compared, by having all systems execute on the same data and the same sets of queries, and as such

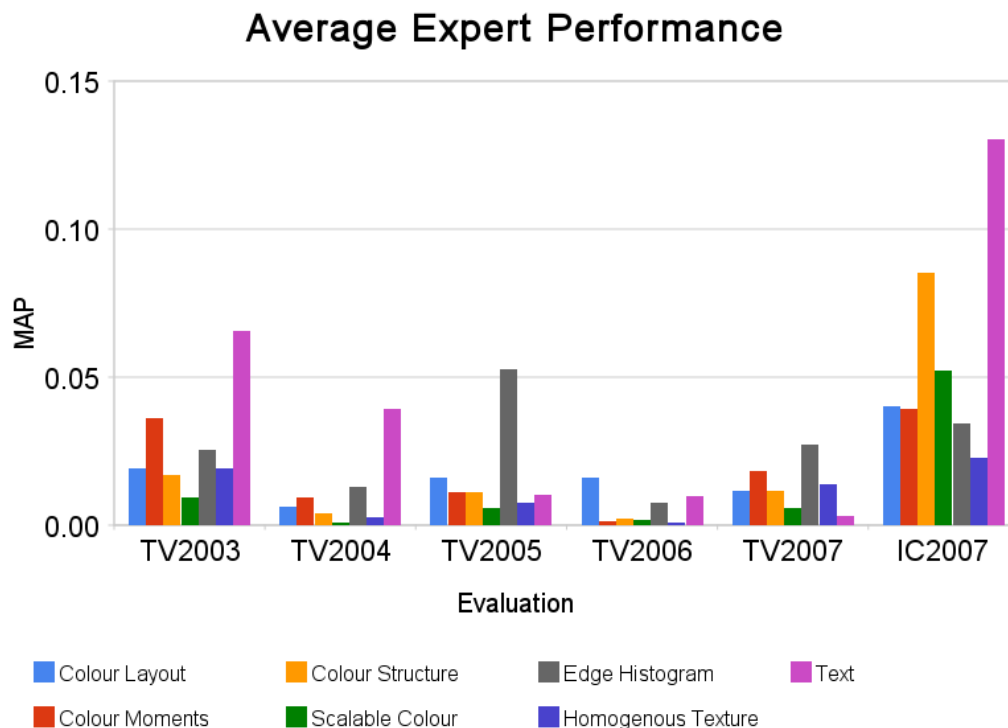


Figure 1.2: Average Individual Expert Performance

have lead to fundamental improvements in the evaluation of search systems. Of course questions remain that systems may become biased towards particular test collections (i.e. overfitting). In this work, we perform our experiments over six different test collections (which we refer to as experimental corpora), in an attempt to generalize our work.

The data on which we are experimenting comes from either collections of digital video, or annotated still photographs. As discussed earlier, we use the phrase ‘document’ to refer to any multimedia artifact used for retrieval. Digital video however introduces a temporal element into the retrieval process, and thus requires conversion into some discrete form to allow for retrieval. The element we use in this work is known as a ‘shot’, which is defined as a visually homogeneous segment of video which is of at least two seconds in duration. From the shot, ‘keyframes’ have been extracted, which are still images which represent the shot.

For our investigation into weighted data fusion for multimedia retrieval, we make extensive use of several benchmarking data collections. Five of these test corpora came from

TRECVID [22] and are digital video collections, and one from ImageCLEF [23], a collection of travel photographs. These two campaigns share similar objectives as both seek to promote research in content-based retrieval by utilising common test collections and open, metrics-based evaluations. Within the five TRECVID corpora however we also have variation, with the data including mono and multilingual video, video from broadcast news and news magazine video. TRECVID also defined the definition of the ‘shot’ and the meaning of extracted keyframes RKF and NRKF, details of which can be found in [22]. The following is a summary of our experimental corpora:

- **TRECVID 2003:** Approx. 60 hours of monolingual English news broadcasts. There are 72,624 total keyframes, of which 37,104 are ‘NRKF’ keyframes and 35,220 are ‘RKF’ keyframes. There are 138 topic images spread across 25 topics. Text evidence is provided through ASR transcripts. Abbreviated to ‘TV2003’.
- **TRECVID 2004:** Approx. 70 hours of monolingual English news broadcasts. There are 48,818 total keyframes, of which 33,367 are ‘RKF’ keyframes and 15,451 are ‘NRKF’ keyframes. There are 160 topic images across 24 topics, and text evidence is provided through ASR transcripts. Abbreviated to ‘TV2004’.
- **TRECVID 2005:** Approx. 80 hours of trilingual news broadcasts in Arabic, Chinese and English, represented as 78,206 keyframes. Of these 45,765 are ‘RKF’ keyframes and 32,215 are ‘NRKF’ keyframes. Topics are represented by 228 topic images, across 24 topics. Text evidence is provided through ASR transcripts for English, whilst for the additional languages the ASR is run through an MT system. Abbreviated to ‘TV2005’.
- **TRECVID 2006:** Approx. 160 hours of trilingual news broadcasts in Arabic, Chinese and English, represented as 146,497 keyframes. ‘RKF’ accounts for 79,848 keyframes whilst there are 66,844 ‘NRKF’ keyframes. There are 169 topic images across 24 topics, text evidence is provided through ASR transcripts for English, whilst for the additional languages the ASR is run through an MT system. Abbreviated to ‘TV2006’.
- **TRECVID 2007:** Approx. 50 hours of Dutch news magazine video, represented as 295,350 keyframes in total. Of these 19,702 are ‘RKF’ images, whilst for this collection we took the aggressive sampling strategy of extracting ‘K-Frames’, which make up the remaining 275,648 images. For the topics there were 205 topic images across 24 topics. The audio for this video was nearly all Dutch, so all text was first detected by ASR which was then run through an automatic Machine Translation (MT) process. Abbreviated to ‘TV2007’.
- **ImageCLEFPhoto 2007:** 20,000 natural still images which form the IAPR TC-12 Benchmark and 180 topic images across 60 topics. Text evidence comes from well-

formed, noise free text annotations which accompany each image. Abbreviated to ‘IC2007’.

By using these multiple corpora, we introduce a degree of generality to our results and observations, as the data collections used in this chapter cannot be characterized as being of a mono-type, as we have observed in the performance of the retrieval experts in the previous subsection.

### ***Experimental Methodology***

Given that the success or failure of any data fusion retrieval system is dependent upon the optimal generation of weights, and the manner in which those weighted documents are combined, it is essential that we know something about what the ideal distribution of weights for a query is, and what is the most effective manner in which to combine these documents.

In order to gain insights into what the ideal form of weighting for CBMIR data fusion is, and indeed how ranked lists should be combined, we need to deviate from the traditional empirical model typically used to evaluate new algorithmic advances. Traditionally we have some form of training data, either topics, data or both, some proposed model which we want to test and some form of parameters which require tuning and a set of evaluation metrics and relevance assessments. Also included in this model is a test set from which final results will be reported. The common sequence of events is that a model is first optimised on training data, then the optimised model is used on the test data. The final result typically reported is the outcome of the evaluation metrics run on the output of the model on the test data, where presumably the model has not been overfitted.

The major problem with the established empirical model is that of evaluation, where what we want to determine is *not* the comparison of competing models and their associated performance, but rather given a maximally performing model, what parameters were associated with it ? In data fusion tasks we will have a range of input sources of evidence, which we will then combine in some manner in order to compute a final response. The fundamental problem is that using the established empirical model, we could evaluate two different fusion systems, and after executing both having first trained on the training collection we can make the observation that system ‘a’ outperforms system ‘b’ by 15%. On the surface this seems fine, system ‘a’ has achieved a good performance improvement over system ‘b’. However, this 15% is a *relative* increase, it is only meaningful when comparing the two systems under observation, and far less important when we do not know when given a fixed set of inputs, what the maximum achievable performance is. For instance, if system ‘a’ scored a MAP of 0.2, but the theoretical maximum attainable given the same inputs was 0.8, then the relative importance of system ‘a’s 15% improvement is diminished as there is clearly room for greater improvement. However, if the maximum was determined to be 0.21, then that 15% improvement is very significant.

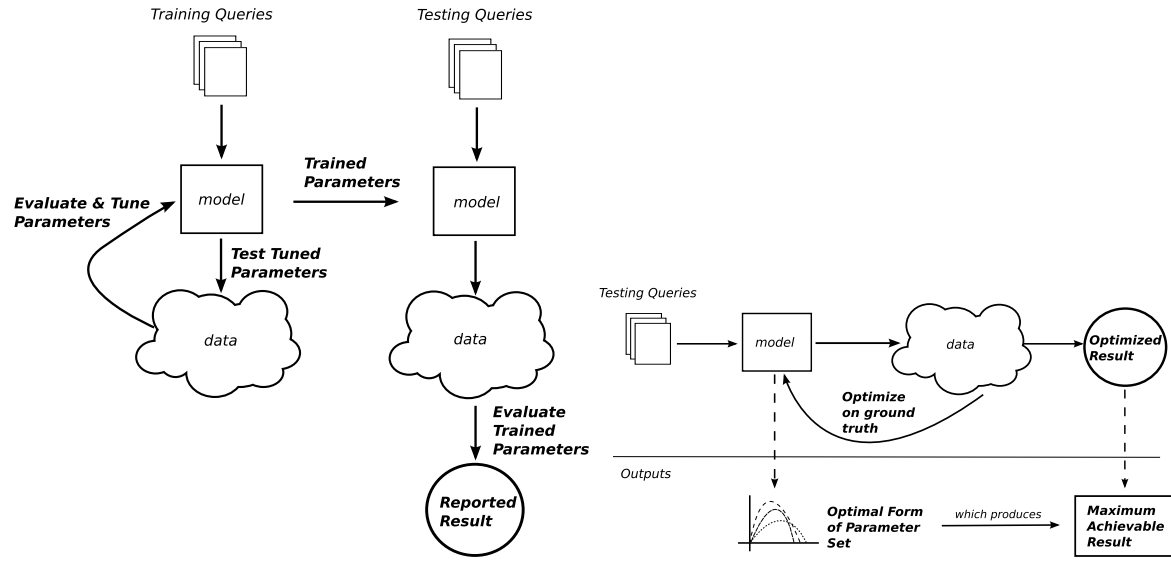


Figure 1.3: Experimental Models, traditional model on the left, our empirical model on the right.

A better use when determining what is the maximum performance for a fixed set of inputs, is to study the properties of this maximally performing model. Rather than being primarily concerned with the maximum performance MAP value, to flip this around such that given we have a model which achieves excellent performance, what are the properties of this model that led to this performance. In order to achieve this it necessitates optimisation directly on the test data. We are in fact not proposing any particular model for data fusion in this chapter, but rather we have created and observed the optimal model for this retrieval problem such that we can report on the properties of this model which future systems should seek to leverage. We illustrate these difference in Figure 1.3, where the traditional model for algorithmic development is on the left, and the model used for these experiments where the objective is to study the optimal model and what parameters comprised it, on the right.

The objective of this chapter therefore is to study what are the variables which generate a maximally performing CBMIR data fusion system, and if there are any commonalities to these that can be discovered so as to inform the development of new data fusion algorithms and systems. The performance of these models is in themselves immaterial, it is these factors which we wish to identify and examine. In particular in this chapter, we examine the impact of combinational hierarchies upon performance, the distribution that an ideal form of weighting takes, the effect of different normalization strategies and the operators used for combining the ranked results. In other words we:



1. establish for any given query, what the ideal form of linear weighting is for that query. This allows us to identify and observe what form of weighting a data fusion algorithm should seek to emulate.
2. determine the capability of finding the ideal set of weights for any given query or set of retrieval factors which allows us to essentially freeze the impact that the weighting scheme has on retrieval performance. Therefore we can robustly test factors such as combination operators, normalisation approaches and combination levels, where for each as the ideal set of weights has been used we can cross-compare results knowing that the performance achieved is the best performance possible using that particular combination of factors.

No doubt at this point after mention of optimizing directly on the test set, alarm bells are ringing in several readers' minds, however we believe we have good justification for doing this. Whilst we have laboured the point as to why we are examining models optimized on the test data, we believe this is necessary as the approach is unorthodox, and inadequately justified can lead to the the results presented in this paper being dismissed out of hand. Furthermore, we emphasize at this point that the results presented here are empirically generated observations as to what constitutes ideally-weighted data fusion. We believe that these observations in several cases are unexpected and should form the foundation of the continued development of data fusion algorithms. This method however is clearly not appropriate for the testing of any new derived algorithm itself, as in so doing such an algorithm would be prone to overfitting.

Fundamental to implementing this experimental framework is the implementation of an appropriate optimisation framework. Several approaches were considered including standard grid searches and statistical approaches such as Expectation Maximisation. We selected the approach known as *coordinate ascent*. This approach was recently adapted for linear combination IR tasks, with direct optimisation on the relevance assessments by Metzler *et al.* [16], in whose work a complete description of the method can be found.

### ***Exploration of Data Fusion Variables***

In this section we discuss and examine weighted data fusion in detail. This section will identify the various components of a data fusion framework, and test each component to determine its impact upon retrieval performance. A difficulty in this explanation is that to test certain attributes, we have to lock down on others before we have explained them. Therefore if the reader is unfamiliar with some of the approaches used, s/he should jump to the appropriate sub-section to familiarise themselves. In each case, where one variable is being tested, the other variables were set to that which maximized performance except where otherwise noted.

The framework we are using for processing a weighted data fusion query is as follows:



1. Issue multi-part multi-expert query (i.e.  $\{query_1 \dots query_j\}$ ) against our retrieval experts (i.e.  $\{expert_1 \dots expert_i\}$ ).
2. For each generated result set,  $rs_{i,j}$ , normalize the documents within that set.
3. Determine what form of a combinational hierarchy is to be used, i.e. will result set be combined into a single set for each expert, for each query component, or will they be directly combined.
4. Based upon the previous step, apply an appropriate weight (either  $rc_i$ ,  $rc_j$  or  $rc_{i,j}$ ) to the candidate result sets.
5. Using the combination operators (either CombSUM or CombMNZ) combine the constituent parts into the final result.

This is a very high level overview of the data fusion operation, and as can be found in an implementation, several loops of this procedure may result in the final computation of a result. The order in which we iterate through these steps in this section is that first, we examine the ideal distribution of weights generated in optimal data fusion. Following this, we examine the impact of combinatorial hierarchies on retrieval performance. Next we examine the impact of the use of either CombSUM or CombMNZ, finishing by examining the impact different normalization strategies, based upon using the score or the rank, have on retrieval performance.

### ***Optimal Distribution of Weights***

In this sub-section we explore the form of an ideal weighting scheme so as to determine if any unique properties exist within it which may guide future data fusion algorithm development. In this experiment, we use score-based normalization through MinMax, with CombSUM for combining results, and we use a direct level of combination. As previously stated our mechanism for doing this is the execution of the coordinate ascent optimisation technique directly on our test corpora and relevance judgements. The first results of this process are presented in Figure 1.4, which is a histogram of the distribution of the ideal weights generated over all corpora. The y-axis represents frequency, whilst on the x-axis, the assigned weights have been transformed into Z-Scores, to allow for cross-comparison between topics and corpora. Z-Scores (also known as standard score) shift and scale values to have a mean score of 0 and standard deviation of 1 allowing us to express a value in terms of how many standard deviations it is away from the mean. Therefore this normalisation allows us to examine how clustered the assigned weights are. For clarity, each *topic's* weights were normalized, meaning for each topic the average weight after normalization is 0. The results of the Z-Score's for each topic are aggregated into the presented graph and data.

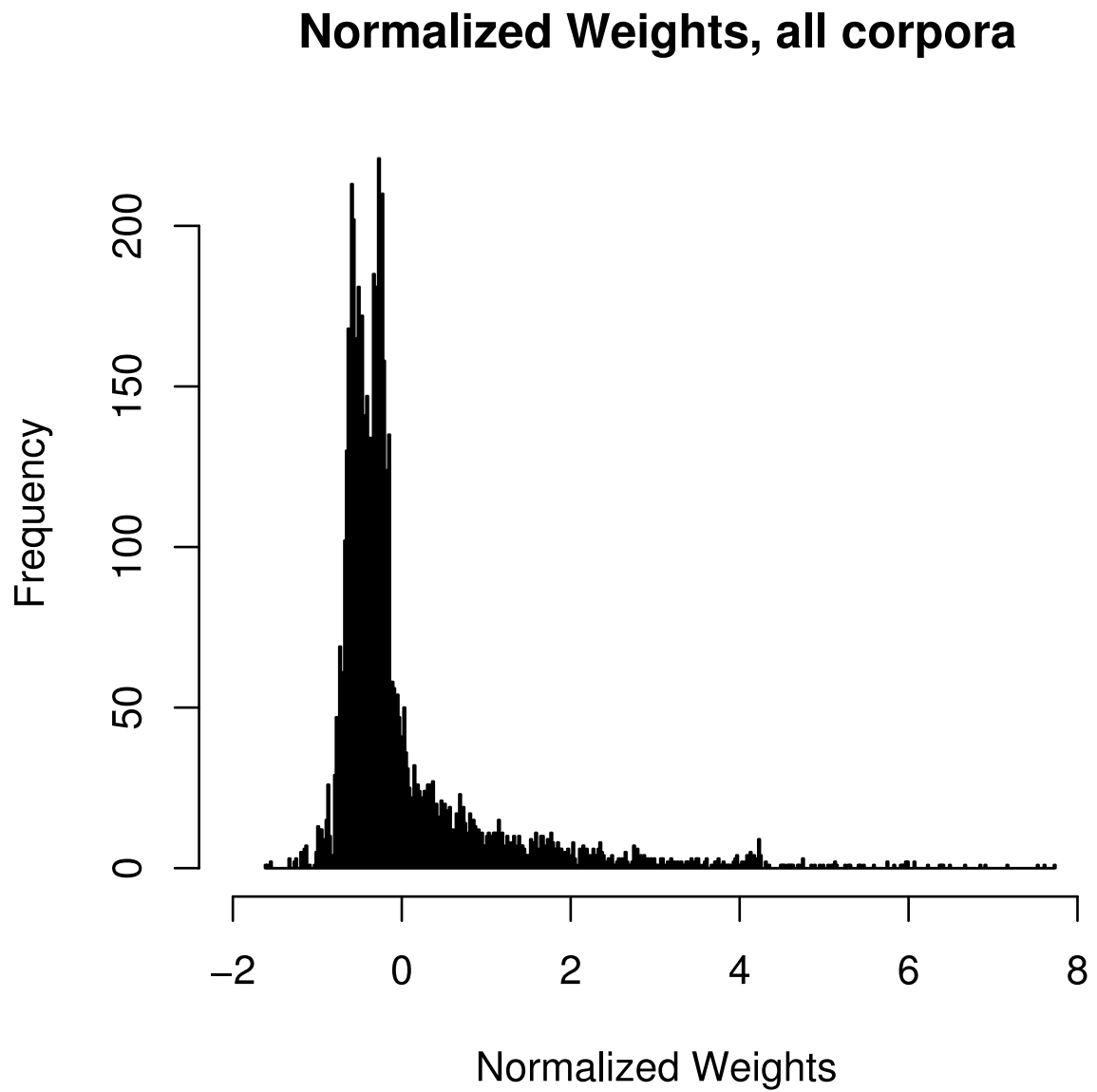


Figure 1.4: Histogram of weight distribution, all corpora

From the histogram we can see demonstrated a highly positively skewed distribution, with a long tail of values extending up to Z-Score values of nearly  $8\sigma$ . The shape of this distribution closely resembles that of a log-normal distribution, characterised by the extended positive tail. To further examine the skewness of the ideal weight distribution, we present various measures of central tendency in Table 1.2. From this we can see that whilst the mean of the weights is approximately 0 and the standard deviation has a value of 1, that the median value is less than the mean value, correlating with our positive skew observation. Furthermore, an examination of the quantiles reveals a very high degree of clustering with an extended tail, as the 75<sup>th</sup> quantile has a value of only 0.0036, whilst it isn't until the 90<sup>th</sup> quantile that values exceed 1.

Mean	Median	$\sigma$	75 Quantile	90 Quantile
0.0004	-0.2978	1.0004	0.0036	1.1192

Table 1.2: Measures of central tendency for ideal weight distribution, all corpora.

This implies firstly, that whilst the distribution of weights has some properties of that of a normal distribution, such as a majority of the data points clustered around the mean and within the range  $\pm 3\sigma$ , there does exist a very definitive positive skew. Secondly, as part of this positive skew, approximately 10%-11% of the weights were assigned values  $> 1\sigma$ . This indicates that a minority of the pairs  $\langle Expert_i, Query_j \rangle$  received the majority of a topic's weight.

The histogram presented is an amalgamation of the weights for all topics over all corpora. Without other evidence there remains the possibility that the effect presented is a corpora-specific event and that the weights are indeed more normally distributed. To account for this we present in Figure 1.5 corpora-specific plots of the weight distributions in the form of quantile-quantile (Q-Q) plots. In each of these figures, the x-axis represents a theoretical normal distribution of weights, whilst the y-axis is the actual weight which was assigned. The dashed line displays the trend line of the weights if they were normally distributed.

Examining each of the six Q-Q plots, we can see that all of our experimental corpora follow the same distributional pattern, as each demonstrates a significant departure from a normal distribution, particularly once the normalised weights values exceed  $1\sigma$ . The pattern shown in each plot is similar to what would be expected if the distribution of the weights was log-normal, again we can also see demonstrated in each plot, a positive skew.

Therefore, the evidence presented suggests that from the optimisation process, the ideal weighting form for data fusion constitutes a majority of pairs  $\langle Expert_i, Query_j \rangle$  being assigned relatively low weights, whilst a handful of select pairs being aggressively weighted. Once again there remains the possibility that this observation whilst corpora-independent may be topic dependent. To explore this, we examined the distribution of weights within

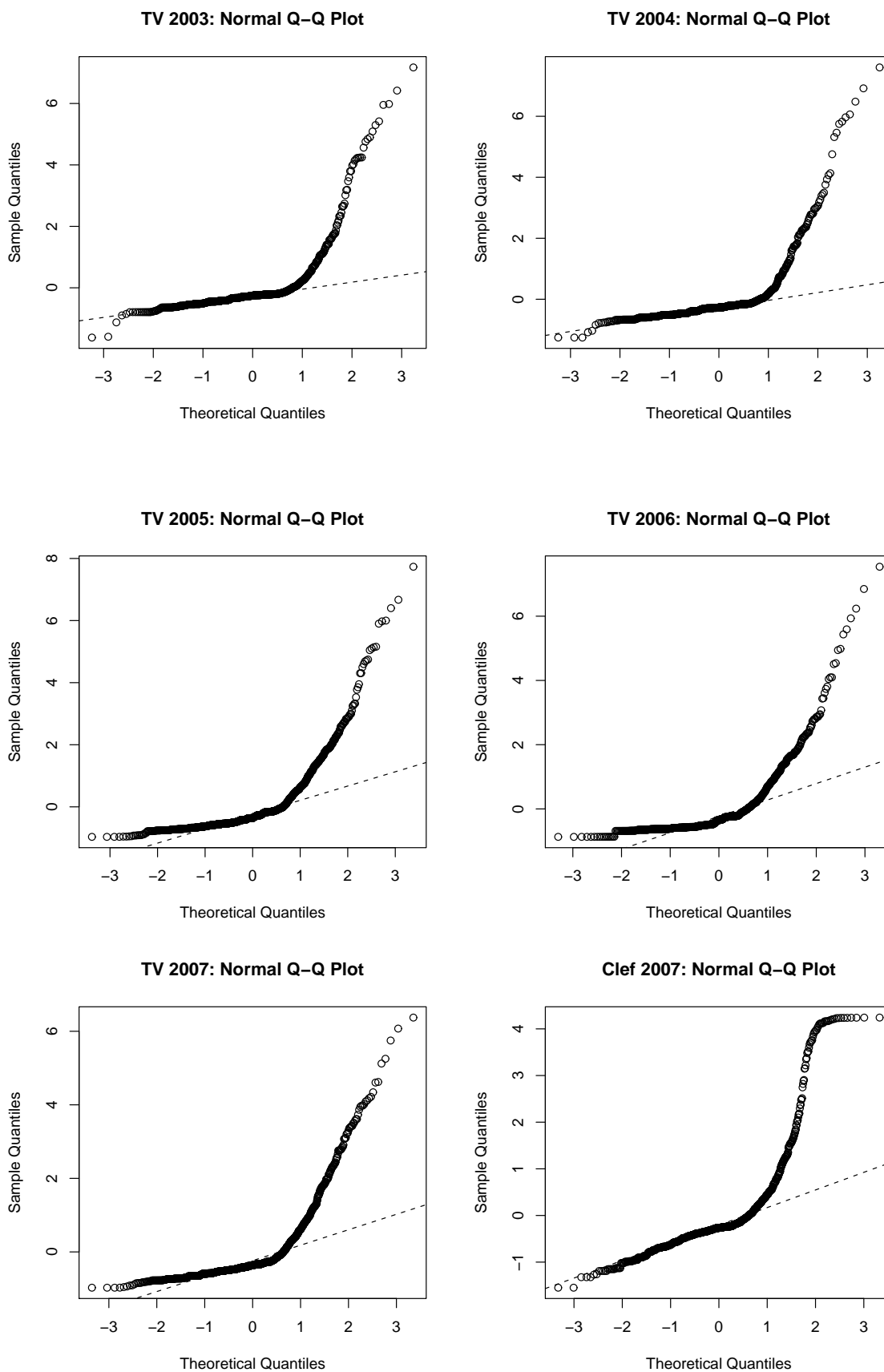


Figure 1.5: Q-Q Plots of Weight Distributions

each topic for each corpora, the results of which are presented in Figure 1.6.

In this figure there is a graph for each of our experimental corpora. We examined for every topic what proportion of pairs  $\langle Expert_i, Query_j \rangle$  were assigned normalised weights greater than  $1\sigma$ , and for these highly-weighted pairs, what the sum of their weights was, or in other terms, what proportion of the total weight did these highly-weighted pairs constitute.

Within each graph, the x-axis represents individual topics for each corpora. For every topic there are two bars, a yellow bar which represents the proportion of pairs  $\langle Expert_i, Query_j \rangle$  which received weight greater than  $1\sigma$ , and a blue bar which demonstrates what percentage of the total weighting did that represent. Taking as an example the first topic in corpora TRECVID 2003, we can see a yellow bar at approximately 16% and a blue bar at approximately 78%. This indicates that for topic ‘0100’ in TRECVID 2003, 16% of the pairs  $\langle Expert_i, Query_j \rangle$  used for that topic received 78% of the total weighting. From these graphs we can see that generally across topics the pattern remains the same, that between 10%-20% of the pairs  $\langle Expert_i, Query_j \rangle$  used for that topic attracted between 60%-80% of the weight. Unlike our previous observations, these graphs were also generated from an optimisation process which used visual-only experts. This was selected so as to further determine if just one modality, i.e. text, was having a significant impact upon the distributions. From these results we can see that the general effect holds, that a minority of pairs attract a majority of the weight. The possibility does exist that even in this case, it may be one particular visual expert which is dominating the weighting and thereby causing the skewed weighting distribution. Table 1.3 shows the breakdown of visual experts and the proportion of the weight assigned. There is a slight bias towards the Edge Histogram and to a lesser extent the Homogeneous Texture experts, however as there are only two texture but four colour experts, this bias can be accounted for. Taken together, the data presented in Figure 1.6 and Table 1.3 shows that highly weighted  $\langle Expert_i, Query_j \rangle$  are distributed across different experts.

CL	CM	CS	SC	EH	HT
15%	17%	13%	13%	24%	18%

Table 1.3: Distribution of Retrieval Experts in  $\langle Expert_i, Query_j \rangle$  with  $rc_{i,j} > 1\sigma$

We have observed that the key to maximising AP is to correctly identify the salient pairs  $\langle Expert_i, Query_j \rangle$  and ensure that these are highly weighted, rather than weighting the overall performance of any given retrieval expert. Therefore the task now is to test this observation to determine how robust it is. We devised a series of experiments that use only the highly weighted pairs  $\langle Expert_i, Query_j \rangle$  to see if we still achieve good performance, or if the remaining lowly-weighted pairs contribute to maximising performance by boosting

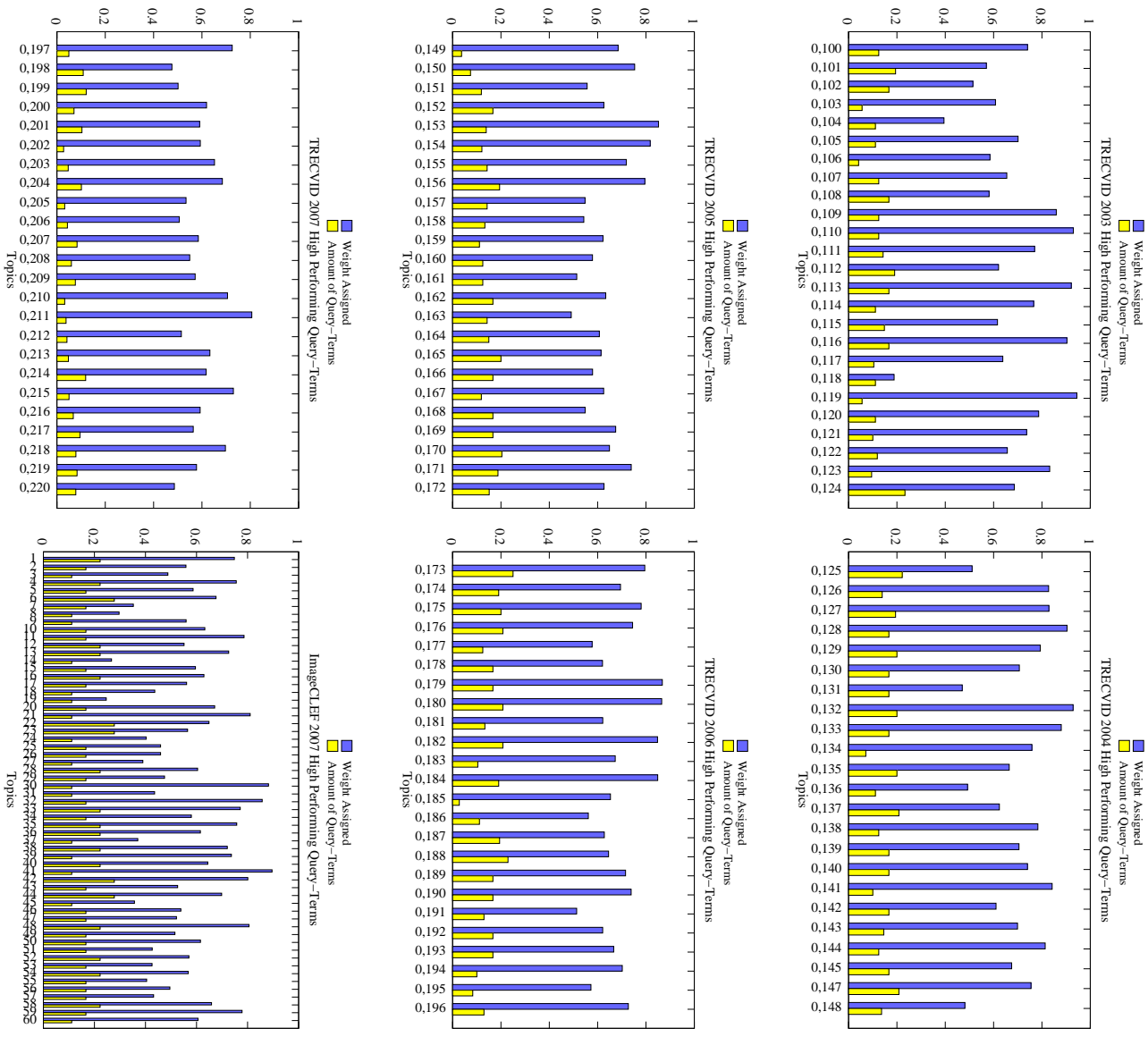


Figure 1.6: Highly Weighted Pairs, all corpora

recall. For clarification, a highly-weighted pair  $\langle Expert_i, Query_j \rangle$  is a pair whose weight  $rc_{ij}$  is greater than  $+1\sigma$  of the mean weight for that topic.

However, before conducting these experiments, we first examine the impact of combination levels on retrieval performance, so that these observations can be tested and contrasted against the combination level experiments.

### Combination Levels

As previously illustrated, the CBMIR system which we are using in our work is capable of handling multiple retrieval experts and multi-example multi-modality queries. We have seen that each unique pair  $\langle expert_i, query_j \rangle$  is able to generate a ranked result set  $rs_{i,j}$ . The objective of the CBMIR system is to combine all of these into a coherent single response. The nature of this setup is that it allows for hierarchical partitioning of the ranked results to facilitate this final combination. That is, systems may employ a multi-stage combination process, either to allow for easier training of weights to populate the weighting matrix  $\mathbf{RC}$ , or because employing a combination hierarchy was accepted practice. For a late fusion CBMIR system, there are three basic levels of combination which could be employed, which we refer to as ‘query level’ combination, ‘expert level’ combination and ‘direct level’ combination. Each of these levels is illustrated in Figure 1.7.

Earlier in our definitions of terminology, we stated that a CBMIR system could handle multi-example multi-modality queries,  $Q$ . The set  $Q$  defined a single search topic which the CBMIR system is to process. The example as given in Figure 1.7, is that our query may be comprised of a yellow flower image and a red flower image and the intent of this query may be to find images of flowers regardless of colour [20]. In the example in Figure 1.7 there are also two visual experts present, a colour expert and an edge expert. Applying our queries to the experts generates 4 result lists to be combined. Using our notation the four result set we have generated are:

$$\begin{aligned} E &= \{colour, edge\} \\ Q &= \{yellowflower, redflower\} \\ \mathbf{RS} &= \begin{pmatrix} colour, yellowflower & edge, yellowflower \\ colour, redflower & edge, redflower \end{pmatrix} \end{aligned}$$

### Expert Level Combination

The first of the three levels we’ll discuss is ‘expert level’ combination. For a specific expert ( $expert_i$ ) we query against it all query components in  $Q$ , merging the results to produce for each expert, a single ranked list. Typically the combination of the individual results from an expert into a single result for that expert are uniformly weighted. Therefore, for every  $expert_i$  we have one combined result set ( $rs_i$ ). The final merger therefore is to combine each

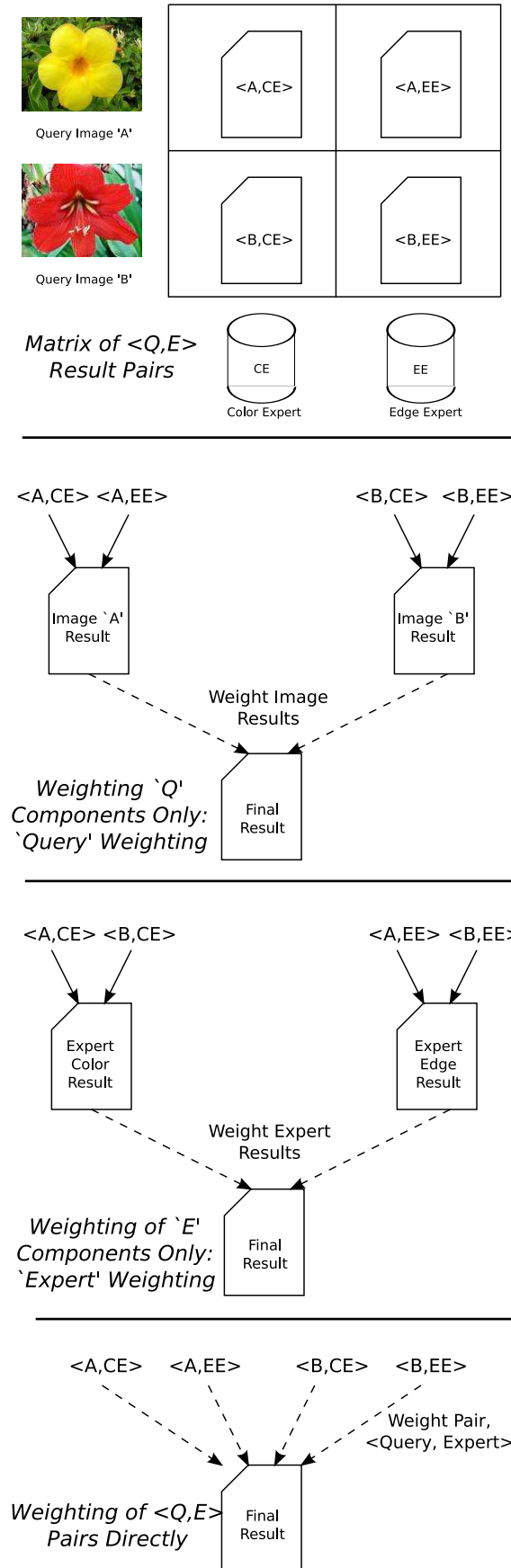


Figure 1.7: Combination levels for single search, with 2 experts (E) and 2 query images (Q), giving 4 ranked lists (pairs  $\langle Q_j, E_i \rangle$ ).



result set  $rs_i$  into a single response. In systems which implement this style of combination, it is at this level of aggregation that weighting would occur, that is each  $rs_i$  would be assigned a weight. This means that the number of weights which must be determined using this level of aggregation is  $|E|$ . To illustrate this with the example from Figure 1.7, the result sets from pairs  $\langle colour, yellowflower \rangle$ ,  $\langle colour, redflower \rangle$  form one merged result set which is then weighted, whilst pairs  $\langle edge, yellowflower \rangle$ ,  $\langle edge, redflower \rangle$  form the other result set to be weighted. As there are two experts ( $|E| = 2$ ), then two weights are used to calculate this query. Referring back to our result set matrix **RS** with individual elements  $rs_{i,j}$ , we can think of this approach as first aggregating the result sets in each column  $i$ , weighting each of these, then combining all instances of the columns  $rs_i$  into a single response.

### Query Level Combination

If ‘expert level’ combination is the processing of the columns of the result set matrix **RS**, then ‘query level’ combination is the same processes except that we instead process the rows of matrix **RS**. Given a set of query components  $Q$  with members  $query_j$ , we combine with uniform weights the results of  $query_j$  queried against every expert in  $E$ . Similarly to ‘expert level’ weighting, the aggregated result set for each  $query_j$  is then weighted and combined to compute the final response to the query. Therefore, each  $rs_j$  is weighted, and the total number of weights used in tuning the system is  $|Q|$ . Illustrating this with our example from Figure 1.7, there are two result sets  $rs_j$ , the first generated from a merger of the pairs  $\langle colour, yellowflower \rangle$ ,  $\langle edge, yellowflower \rangle$ , and the second from the pairs  $\langle colour, redflower \rangle$ ,  $\langle edge, redflower \rangle$ .

### Direct Level Combination

Finally we have the “direct” level of combination, where if the ‘expert level’ of combination was the aggregation and weighting of columns  $rs_i$ , and ‘query level’ combination was the aggregation and weighting of rows  $rs_j$ , then ‘direct level’ is the direct weighting of each individual result set  $rs_{i,j}$ , in other words, processing the matrix **RS** directly without any intermediate levels of aggregation. This level specifies weights for every coupling of a query component and retrieval expert, meaning that using this approach,  $|E| \times |Q|$  weights are required. Applying this to our example from Figure 1.7, this means that the four pairs  $\langle colour, yellowflower \rangle$ ,  $\langle edge, yellowflower \rangle$ ,  $\langle colour, redflower \rangle$  and  $\langle edge, redflower \rangle$  each have their own weight.

### Experiments with Highly Weighted Pairs and Combination Levels

To test our observations we devised three experiments in order to (1) determine to what extent the highly-weighted pairs  $\langle Expert_i, Query_j \rangle$  impact upon performance; (2) determine if the weighting of these pairs needs to be exact or if merely identification is enough;

and finally, (3) determine the impact the remainder of the pairs  $\langle Expert_i, Query_j \rangle$  which do not have much weight allocated to them have on performance. As a comparison we have also included two optimizations, “Query” level and “Expert” level optimizations. These represent the best performance achievable if we use existing data fusion methods (such as Query-Class approach, or single feature machine learning [24],[25]), and allow us to determine if our suggested strategies of targeted weighting of pairs  $\langle Expert_i, Query_j \rangle$  rather than expert level weighting offers improvement.

- **(1 $\sigma$ ) 1 $\sigma$ :** For each topic, only use highly-weighted pairs  $\langle Expert_i, Query_j \rangle$  (i.e. pairs  $\langle Expert_i, Query_j \rangle$  whose assigned value from optimization was +1 $\sigma$  for the mean weight). The value of  $w_{ij}$  will be the value determined during optimization. This test will examine the impact of precisely weighted high-performing pairs  $\langle Expert_i, Query_j \rangle$ . It can be thought of as a high-precision experiment as for each topic we will be using only 5%-20% of the available ranked lists for that topic.
- **(1 $\sigma$ U) 1 $\sigma$  Uniform:** Using only the highly-weighted pairs  $\langle Expert_i, Query_j \rangle$ , assign each a uniform weight. This will examine if just the identification of high-performing pairs  $\langle Expert_i, Query_j \rangle$  is sufficient to yield performance increases, specifically determining if accurate weighting of pairs is required, or if they can be assigned a binary weight [0,1]. As the task of determining the optimal set  $w_{ij}$  is realistically only viable post-experiment, this experiment tests if realistic fusion approaches can be developed, as it does not require perfect weights, only identification of likely high performing pairs  $\langle Expert_i, Query_j \rangle$ .
- **(1 $\sigma$ U-T) 1 $\sigma$  & Tail:** We extend experiment 1 $\sigma$ , by taking the remaining weight mass that isn’t assigned to high-performing pairs and allocate it uniformly amongst the remaining pairs in **RC**. This experiment complements the previous, we assign a large weight to the high-performing pairs, whilst a low weight to the remainder. As the high-performing pairs constitute only 5%-20% of available pairs for a topic, this experiment is testing the impact of recall, i.e. can we include the remainder of the data without accurate weighting so as to increase our recall.
- **Expert Optimized:** We implement the “Expert” level of combination as is implemented by several data fusion approaches. Here we use the optimization approach as previously described so as to determine the near-optimal set of weights  $RC_i$  for “Expert” level combination. This demonstrates the best performance that can be expected using the same query images and experts as the previous experiments if we impose a combination hierarchy at the “Expert” level.
- **Query Optimized:** This experiment is as for “Expert” optimized, except that the weight set we are optimizing is  $RC_j$ , and demonstrates the best performance that can be achieved if we combine at the “Query” level.

For each experiment we include the minimum and maximum achieved for that corpus. The minimum is a ‘Uniform’ run, where all pairs  $\langle Expert_i, Query_j \rangle$  are equally weighted, demonstrating the performance achieved if no weighting scheme at all is employed. The maximum is the fully optimized result, demonstrating the best performance that can be theoretically achieved. These two figures provide a lower and upper bound for data fusion performance comparisons, allowing us to make decisions using absolute observations with regard to the bounds, rather than relative observations by comparing only to existing data fusion approaches.

Our results are presented in Figure 1.8. Each table presents the minimum (Uniform All), maximum (All Optimized) and results of the 5 experiments using MAP. For every experiment’s MAP, we show in brackets how close that approach came to achieving the optimal performance. The MAP of each of the experiments, along with the maximum MAP, is graphed in Figure ???. For each of our 5 runs we ran significance tests (partial randomization) with  $\rho$  0.05. For the TRECVID benchmarks we found no significant difference between the ‘Query’ and ‘Expert’ levels of hierarchical combination, indicating that if hierarchical combination is employed and optimally weighted, there is no difference in between them. (However for ImageCLEF ‘Expert’ was significantly different.) For benchmarks TRECVID 2003 to 2006, all runs using highly weighted ( $1\sigma$ ) pairs performed significantly better than the hierarchical combination approaches. For TRECVID 2007, only run  $1\sigma$  was significantly different.

The graph presents a clear stratification of the results, particularly for benchmarks TRECVID 2003 to 2006. We can clearly see the very large discrepancy in performance between the hierarchical fusion approaches (at the bottom of the graph) versus the targeted weighting approaches in the middle. This separation illustrates the performance gains achievable by moving away from hierarchical combinations. Of exception is TRECVID 2007 and ImageCLEF 2007, where there is less of a difference in performance. These two benchmarks exhibit the greatest ratio of topic images to collection images – in the case of ImageCLEF one topic image for every 112 collection images. This indicates that recall plays a more prominent role in these evaluations, and that the selection of highly-weighted pairs may have been too restrictive to provide adequate topic coverage. This is reinforced by the run  $1\sigma$ U-T, which included all pairs  $\langle Expert_i, Query_j \rangle$ : it performed the best even though it used non-specific weights.

The run  $1\sigma$  highlights that using a subset of the highly weighted pairs  $\langle Expert_i, Query_j \rangle$ , very good performance can be achieved despite a reduction in potential recall by not using all pairs. Far more encouraging is the performance of runs  $1\sigma$ U and  $1\sigma$ U-T. Whilst run  $1\sigma$  had value as an illustrative run, it is hard to conceptualize a data fusion algorithm that would create the exact optimal weights for these pairs. However, as runs  $1\sigma$ U and  $1\sigma$ U-T did not use the optimal weights, but rather only identified what the high-performing pairs  $1\sigma$ U and  $1\sigma$ U-T were (essentially a binary weighting), and still achieved excellent performance, it provides a clear direction for development of data fusion algorithms. These runs demonstrate that if methods can be developed to identify pairs  $\langle Expert_i, Query_j \rangle$  that are

Legend	TRECVID 2003	TRECVID 2004	TRECVID 2005
Run	MAP	MAP	MAP
Uniform All	0.0593	0.0288	0.0646
Expert Level	0.0752 (61%)	0.0519 (47%)	0.0827 (59%)
Query Level	0.0776 (63%)	0.0543 (50%)	0.0850 (60%)
$1\sigma$ Uniform	0.0966 (79%)	0.0738 (68%)	0.1037 (74%)
$1\sigma$ Uniform & Tail	0.0958 (78%)	0.0764 (70%)	0.1109 (79%)
$1\sigma$	0.0989 (80%)	0.0770 (71%)	0.1108 (79%)
All Optimized	0.1224	0.1084	0.1407

Legend	TRECVID 2006	TRECVID 2007	ImageCLEF 2007
Run	MAP	MAP	MAP
Uniform All	0.0164	0.0422	0.1283
Expert Level	0.0299 (53%)	0.0655 (56%)	0.1648 (76%)
Query Level	0.0262 (47%)	0.0700 (60%)	0.1544 (71%)
$1\sigma$ Uniform	0.0460 (82%)	0.0680 (58%)	0.1404 (65%)
$1\sigma$ Uniform & Tail	0.0453 (80%)	0.0762 (65%)	0.1715 (80%)
$1\sigma$	0.0496 (88%)	0.0772 (66%)	0.1439 (68%)
All Optimized	0.0563	0.1175	0.2156

Figure 1.8: Experimental Results, all corpora

likely to be highly weighted, then exact weighting is not required to obtain performance superior to that of methods which employ hierarchies.

### *Equivalence Transformations*

In this sub-section we examine the role of equivalence transformations and their impact on retrieval performance. To re-cap, equivalence transformations, or more generally ‘normalisation’, can be defined into two broad classes, score- and rank-based transformations. The objective of any of the normalisation approaches examined is to perform transformations on the set of  $rs_{i,j}$  which are being combined, such that the effects of different factors like score ranges, numbers of retrieved results or score distributions does not adversely impact upon performance. Recall that in our candidate result sets to be combined ( $rs_{i,j}$ ), each is comprised of documents  $m$ , where  $document_m \mapsto (name, rank, score)$ . Score-based normalisation approaches alter the scores of documents to be combined, thereby allowing combination to occur utilising scores, whilst conversely rank-based transformations alter the ranks such that the rank of a document is what is used for combination.

**Z-Score** Score-based transformation converts the score of a document into the Standard Score [26] (Z-Score) within the  $rs_{i,j}$  from which it came, the Standard Score being a

measurement of how many standard deviations a score is from the mean score. This approach has no range restriction [27, 28].

$$Norm_{score(x)} = \frac{score_x - \mu}{\sigma} \quad (1.4)$$

**Min-Max** MinMax normalisation considers the best and worst scoring documents of a given  $rs_{i,j}$  and assigns these scores of 1 and 0 respectively. Scores are then normalised within the range [0:1] [30].

$$Norm_{score(x)} = \frac{Score_x - Score_{min}}{Score_{max} - Score_{min}} \quad (1.5)$$

**Borda** Borda ranked based transformation, given a set size of  $N$  the Borda transformation assigns a score of  $N - x$ , where  $x$  is the rank. As  $N$  is the size of the result set being transformed, it produces lower scores for result sets which do not contain many documents [19, 29, 27].

$$Norm_{score(x)} = N - rank_x \quad (1.6)$$

**BordaMAX** Extends Borda, where  $N$  becomes the value of the size of the largest result set being combined, which is then used for all result sets. This is to discount the bias against small result sets present in the standard Borda approach.

**RankMM** MinMax normalisation based upon ranks. Conceived as a middle ground between Borda and BordaMAX, RankMM normalises the range of ranks between [0:1].

**Reciprocal** Reciprocal rank is a rank transformation which is heavily biased towards the top ranked documents. The rank transformation is  $\frac{1}{rank_x}$

For these experiments, documents were combined using CombSUM, CombSUM being a linear combination operator. With the exception of the Z-Score approach, all approaches combined both text and visual experts. Because the Z-Score approach does not perform any range restriction, we restricted this approach to visual expert only combination so that result set sizes would not be a factor. The results of our experiment are presented in Table 1.4.

From these results we can infer that rank-based approaches generally outperform any of the score-based approaches. The Z-Score method is a distinct failure with these data sets, and is excluded from further analysis. Likewise for the rank-based approaches, the Reciprocal method performs poorly, although nowhere near as bad as the Z-Score approach. The failure of the Z-Score approach is likely due to the lack of any range restriction of the scores being combined. This would have made the optimisation process problematic, as there would be no equivalence between the documents to be combined, meaning that any weight generated would be required to factor this in. In the case of the reciprocal method,

	Score-Based		Rank-Based			
Year	Z-Score*	Min-Max	Borda	BordaMAX	RankMM	Reciprocal
TV2003	0.0095	0.1958	0.2127	0.2232	0.2206	0.1462
TV2004	0.0040	0.1524	0.1655	0.1774	0.1673	0.0892
TV2005	0.0117	0.1554	0.1654	0.1668	0.1670	0.1069
TV2006	0.0089	0.0846	0.0962	0.0989	0.0969	0.0645
TV2007	0.0195	0.1338	0.1433	0.1562	0.1459	0.0808
IC2007	0.0093	0.3148	0.3480	0.3678	0.3491	0.2503

Table 1.4: Normalisation Results Comparison. n.b.\* values for approach Z-Score are based on visual experts only.

its failure, whilst less severe than that of the Z-Score approach, highlights that the in-built aggressive weighting of the top ranked documents of any result set was too severe. Given that earlier we established that the ideal weights for expert combination take the form of a log-normal distribution, the optimisation process would have had to devise a weight which appropriately scaled the result sets to be combined whilst discounting against the heavily weighted top documents of each result set.

From our remaining four approaches, MinMax and the three Borda variants, we can see demonstrated that the score-based approach of MinMax is outperformed by all of the Borda based approaches. To compare the three Borda variants, we can observe that in terms of performance the standard Borda approach is the worst performer of the three. Given that the text expert often would not have returned the full 1,000 results, the standard Borda approach would have penalised this evidence. In terms of significance, the standard Borda approach is significantly worse than the BordaMAX approach, whilst for TRECVID corpora 2003-2006 there is no significant difference between it and the RankMM approach.

The BordaMAX approach appears to generally be the best performer, however there is no statistical difference between it and the RankMM approach for TRECVID corpora 2003 to 2006. Indeed in the case of TRECVID 2005 RankMM claims the top result. The main interpretation of this result is that rank-based approaches produce very successful results, but should not penalise result sets being combined which are of smaller size.

Whilst not a bad performer in itself, across all collections the MinMax approach is significantly worse than all of the Borda based approaches (with the exception of the standard Borda approach for TRECVID 2003, however in raw performance terms standard Borda is still superior). We note that the combination of results in this experiment used CombSUM. The question of why the score-based approach is less successful than the rank-based approaches we believe is due to the non-linear nature of scores and the application of linear weighting.

Given a result set  $rs_{i,j}$ , the sorted scores of this set are unlikely to exhibit a linear progression, and as such introduce a large degree of variability between the relative performance of individual documents. That is, given any two adjacent ranked documents in  $rs_{i,j}$  the difference between the scores will vary considerably between pairs, whilst for ranks the difference between pairs will always be constant. Once a linear weight is then introduced, the impact of this weight will vary dependent on the value of the score to which it is being applied, whereas for rank approaches the impact of the weight is constant and predictable. This effect has been previously identified by [7], which termed the “independent weighting effect” which effectively introduces a second weight for combination. Upon reflection we can see that this observation is equally relevant for both the Z-Score and reciprocal approaches. In the case of the Z-Score approach, the independent weighting effect is present in the variation of the range which scores can take, whilst for the reciprocal approach it is evidenced in the aggressive weighting of highly ranked documents.

The question of the use of score or rank for data fusion is one which is not always fully considered by researchers. Many papers suggest that the use of scores is better than rank as they provide more information, such as the distribution of values or other variables [27], [31], [32], whilst many others would provide no justification at all, as described by Lee [7]. One reason for this is that much of the work in data fusion, particularly in the text domain, has been the non-weighted combination of similarly performing retrieval experts, where the scores (similarity values) once normalised have a degree of cross-comparability, as the experts from which they came are using similar retrieval techniques on the same types of index [10]. CBMIR alternatively combines experts of wildly varying performance from completely different indexing representations, and as such perhaps too much value is inferred onto the benefit of using scores from research experience found in text IR applications. Indeed as authors such as Robertson [33] and Dwork *et al.* [29] note that the score in many cases is just an artifact which is used for generating the ordering of a ranked list, the value of a score itself beyond this function is meaningless.

In Lee’s work on data fusion he hypothesized that rank combination rather than score combination should perform better, given that scores are impacted by the “independent weighting effect”. However experimentally he found that this is not the case [7], his work using normalised scores provides the better results. Croft[32] provides an interpretation of Lee’s result:

“This can be interpreted as evidence that the normalised score is usually a better estimator for the probability of relevance than the rank. Using the ranks is a more drastic form of smoothing that appears to increase error except when the systems being combined have very different scoring characteristics” [32].

This interpretation fits exactly to the characteristics of CBMIR, where as previously established we have very noisy sources of evidence being combined which vary wildly in performance.



### ***Combination Operators***

Once the results from multiple ranked lists  $rs_{i,j}$  have been normalised, they can then be combined to form a single ranked result for a given information need. In our discussion of the combination operators that can be used, we are assuming that one of the previously discussed transformations have been applied. If this is the case, then the combination approaches which we will now discuss can be applied to either the scores of a ranked list or to the rank of a ranked list. For convenience we will refer to a document's score when discussing combination approaches, however the normalised rank can be easily substituted and take the place of a score. The linear weighting of ranked lists and their subsequent combination has been proposed several times, with early work in linear weighting completed by Vogt *et al.* [34].

The more common approaches to combination of ranked lists is to use variants of linear interpolation. Fox and Shaw[30] in early work defined six approaches for linear combination, of which two have seen the most success, CombSUM and CombMNZ.

#### ***CombSUM***

CombSUM is defined as the weighted sum of a documents score's in each of the result lists in which it appears. For instance, if we have our multi-part query  $Q$  and set of retrieval experts  $E$ , we will generate the matrix of result sets (ranked lists) **RS** and have along side it our weighting matrix **RC**. Given a document  $x$ , we examine each individual ranked list,  $rs_{i,j}$ , obtain the score of document  $x$  and apply the weight  $rc_{i,j}$  to this score. The final score for document  $x$  therefore is the sum of each weighted instance of  $x$  occurring in the matrix **RS**. Formally this is described in Equation 1.7.

$$CombSUM(score_x) = \sum_{rs_{i,j}}^{RS} (score_x \in rs_{i,j}) \times rc_{i,j} \quad (1.7)$$

#### ***CombMNZ***

An alternative implementation developed by Fox *et al.* [30] is CombMNZ. CombMNZ extends CombSUM by introducing a variable which heavily weights documents that appear in more than one result set. This variable is the number of times that a document appears in result sets, i.e. the number of non-zero entries that a document has in the set of ranked lists to be combined (hence MNZ). In our terminology, this equates to the number of times a document  $x$  appears in a ranked list ( $rs_{i,j}$ ) in our result list matrix **RS**. We define weighted CombMNZ in Equation 1.8, where the weight has already been incorporated as part of the calculation of CombSUM.

$$CombMNZ(score_x) = CombSUM(score_x) \times \alpha \quad (1.8)$$



	CombMNZ		CombSUM	
Year	Score	Borda	Score	Borda
TV2003	0.1720	0.1458	0.1958	0.2127
TV2004	0.1321	0.1258	0.1524	0.1655
TV2005	0.1329	0.1230	0.1554	0.1654
TV2006	0.0770	0.0701	0.0846	0.0962
TV2007	0.1157	0.1131	0.1388	0.1433
IC2007	0.2991	0.2861	0.3148	0.3480

Table 1.5: CombSUM vs. CombMNZ for score &amp; rank-based approaches.

where  $\alpha$  is the number of result sets in which a document was found, and  $1 \leq \alpha \leq |\mathbf{RS}|$ . Similar to some of the previous tools that we have examined, the choice of CombSUM versus CombMNZ is down to the intention of the system designer and the type of behaviour they would like present in the search system. CombMNZ aggressively promotes documents which occur in a majority of ranked lists to be combined, whereas CombSUM can be seen as a linear addition.

For this experiment, we will investigate the impact of CombSUM and CombMNZ on both score- and rank-based normalisation approaches. To test the score-based approach we use the MinMax score normalisation technique. For rank we use the standard Borda approach. The experts used for combination involve both text and visual experts and result sets  $rs_{i,j}$  are truncated to 1,000 results. The results of this experiment are presented in Table 1.5.

There is little ambiguity in the presented results as we can quite clearly observe a constant ordering across all corpora with regards to performance. CombSUM using Borda normalisation is the most effective combination operator, whilst CombMNZ using ranks is consistently the worst performer.

However there is one curious artifact, which may explain the preponderance of data fusion literature which advocates the use of CombMNZ. From our results we observe that when scores (MinMax) rather than ranks are used, CombSUM achieves greater performance than CombMNZ once fully optimised. When CombMNZ is considered in isolation, score-based CombMNZ consistently outperforms rank-based CombMNZ, reversing the observation found with CombSUM. This result is of interest particularly when coupled with our previous observations about the popularity of using scores for data fusion in the research literature [7, 32]. In terms of significance, we found that for all instances, CombSUM performed better than CombMNZ with statistical significance.

Of interest however is this result of CombSUM and CombMNZ, particularly given the popularity that CombMNZ holds in data fusion, whilst in our experiments performing empirically worse. Lee’s motivation for demonstrating the effectiveness of CombMNZ was

Doc.	E1	E2	E3	CombSUM	CombMNZ
a	0.8	0.0	0.9	1.7	3.4
b	0.6	0.3	0.6	1.5	4.5
CombSUM ranking				a > b	
CombMNZ ranking				b > a	

Table 1.6: CombSUM vs CombMNZ behaviour

an extension of the work of [6] where they observed that a document which appears in multiple ranked lists should be ranked higher. Lee saw CombMNZ as a mechanism which gave a boost in ranking to documents which appeared in more lists.

Thinking of CombMNZ in terms of providing a boost to documents which appear in multiple lists gives an impression of a function which is positive, that it promotes documents up a ranked list. An alternative way however when thinking of CombMNZ is negative, that in fact the function's purpose is to *penalise* documents which do not appear in *all* ranked lists. We can illustrate this with a toy example given in Table 1.6.

In this table we have two documents, 'a' and 'b', and three experts 'E1 ... E3'. Document 'a' is only found in two of our three experts, however it is highly ranked positions in both of these experts. Document 'b' on the other hand is found in all three experts, but it appears around the middle of each experts result set. The combination behaviour of CombSUM and CombMNZ in this case produces two quite different orderings. CombSUM will rank document 'a' before document 'b', whereas CombMNZ will reverse the ranking putting 'b' before 'a' as it appears in all three of our result sets. The question for system builders becomes what sort of behaviour do they want present in their retrieval system ?

We believe that the CombSUM behaviour is more desirable, particularly considering that once any large number of result sets are to be combined through data fusion, that the probability of a document being found in every, or a majority of result sets, must decrease. This has been empirically justified in the results we have presented in this chapter for CombSUM and CombMNZ.

### ***Discussion***

In this section we explored the different ways data fusion can be implemented when combining multi-part queries from multiple retrieval experts. We observed that the best levels of performance can be achieved when specific instances of  $rs_{i,j}$  can be identified and given greater weight. Two important caveats need to be mentioned at this point. Firstly, whilst we identified this attribute, the development of any technique to identify these instances of  $rs_{i,j}$  at query-time is a non-trivial and significant research problem. Secondly, there remains the possibility that these observations may be more pronounced with the experimental corpora used in this investigation, specifically the news video collections. We could

speculate that performance is more pronounced with these collections because there is a degree of redundancy in these collections (attributes such as advertisements), which may be having some form of an impact in our observations. We cannot discount this without further experimentation on more diverse corpora, however we believe that by using the six experimental corpora that we have, that a strong case can be made at the very least for further robust investigation.

### *Current Approaches for Weight Generation and Data Fusion*

Here we present related work which currently addresses the data fusion problem and weight generation for multimedia applications. However, before proceeding we mention of two previous studies which have examined several of the issues we presented in this chapter, specifically work by McDonald and Smeaton [31] and by Yan and Hauptmann [35]. McDonald and Smeaton empirically compare combination approaches for score, rank and probability techniques. Their work evaluated these approaches by optimizing Mean Average Precision (MAP) on a training collection with multiple topics, then applying these generalized optimized parameters to a test set. Our work differs as our optimizations occur at the topic level (Average Precision), rather than the topic set level (MAP). Furthermore the fusion approaches detailed in [31] use a hierarchical approach which is likely to obscure the effect individual query images have on performance. Yan and Hauptmann [35] conduct experiments with TRECVID 2002 data to construct a theoretical framework for studying the upper bounds of combination functions. They found that linear forms of combination may be too restrictive for large numbers of experts to be combined effectively. However, like McDonald and Smeaton, this work examined combination at the expert level and as such did not delve down to the granularity of pairs  $\langle expert_i, query_j \rangle$ . Whilst there could be little argument that linear weighting could not be improved upon by non-linear weighting, the evidence we have presented in this chapter which uses linear weighting certainly indicates that very good performance could be extracted through linear weighting schemes.

### *Query Independent Weighting*

Query Independent Weighting is one of the simplest methods that can be employed when combining retrieval experts. Popular in earlier research in CBMIR [17, 18, 19] because of its simplicity, query independent weighting is an empirical method which requires either a training corpus or domain knowledge of the collection being indexed. Weights are statically assigned for each expert in the system and do not change, regardless of the query being issued. For all queries being processed by the system, this single weight matrix, which weights only experts, is always used. Whilst relatively simple to execute, this method is only suitable when indexing data from a narrow domain [12].

### **Query-Class Weighting**

*Query-Class* weighting can be seen as an evolution of query independent weighting as it directly addresses many of the failings of query independent weighting and as such has proven popular in the multimedia community. The central concept of this approach is that given a training collection and an appropriate set of training queries, query clusters (i.e. *query-classes*) can be found such that queries within each cluster share some similar ‘properties’ which differentiate them from other queries in the collection (where ‘properties’ may be artifacts such as semantic similarity, performance similarity, distance etc). By partitioning a set of training queries into discrete *query classes*, it is then possible to optimize for each *query-class* an instance of the weighting matrix  $\mathbf{RC}$ , such that each class should have a different set of weights for combining retrieval experts. When a test/live query is then processed, it is first mapped to a *query-class* so that the relevant matrix  $\mathbf{RC}$  can be applied for the retrieval experts used. As such, this provides a considerable improvement over query independent weighting, notably in the granularity of how experts can be weighted as every *query-class* has an associated weight matrix  $\mathbf{RC}$ . Whilst the query independent approach only has one instance of  $\mathbf{RC}$  to weight queries, in the *query-class* approach there are  $|\text{query-classes}|$  available for weighting. However like the query independent approach, our weighting matrix only weights at the expert level, not at the level of  $\langle \text{expert}_i, \text{query}_j \rangle$ , therefore we refer to these matrices again as  $\mathbf{RC}_i$ . Figure 1.9 provides a high level overview of the *query-class* approach.

The *query-class* approach was initially developed by [36] closely followed by [37] for content-based video retrieval on the TRECVID corpora. Both share a degree of similarity in implementation as both statically define *a priori* the *query-classes* to be used by the system. [38] extend the initial approaches by automatically discovering *query classes* in a training set, free of any manual involvement. Their approach is based on the observation that the intent of *query class* weighting is to group queries together such that they share the same set of expert weights to achieve good performance. Therefore Kennedy *et al.* hypothesize that given a set of retrieval experts and training queries, that those queries which share similar performance variations between experts (where performance is measured by an evaluation metric, i.e. average precision) should belong to the same *query class*. One of the latest evolutions in the *query class* approach is provided by [53], whose work concentrates on creating dynamic *query-class* weighting during system operation, rather than the previously described approaches which learned classes from training data.

### **Discriminative Classification**

Machine learning applications can be considered as techniques which discover structural patterns in data that allow us to explain and make predictions from that data [39]. They provide us with tools to help make sense of and to organize data. The field of machine learning is incredibly broad and a general review is outside the scope of this chapter. In this section, we restrict ourselves to a brief review of approaches which can be characterized

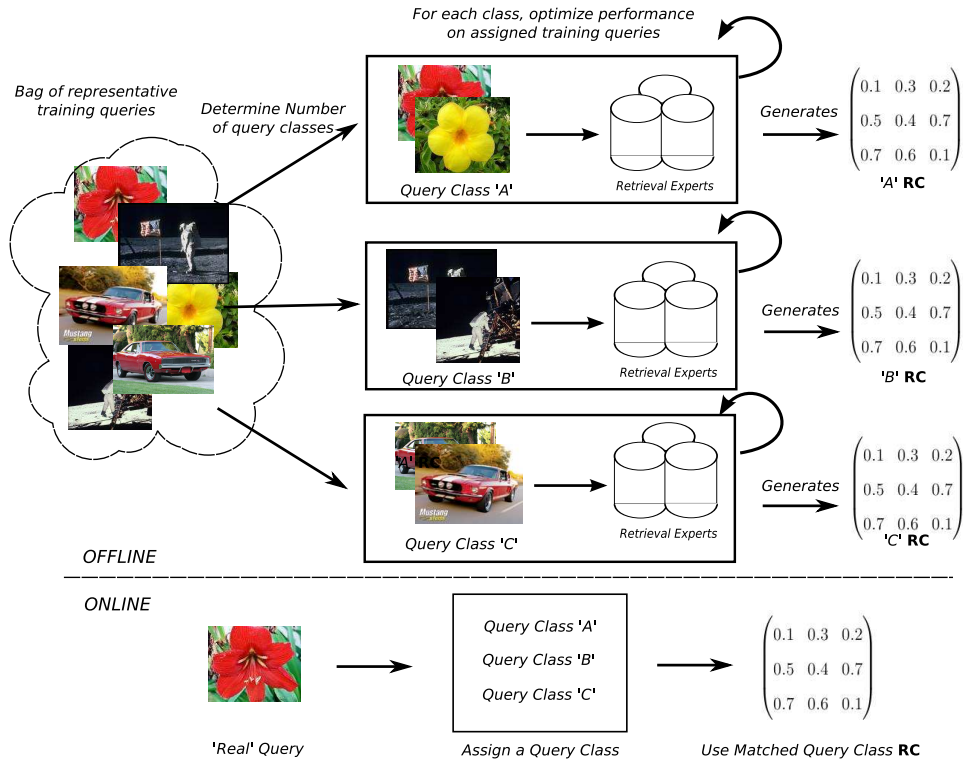


Figure 1.9: Query Class Weighting

as *supervised* and *discriminative*, that is approaches which require training data which has been manually labelled, and models which learn directly on the data provided to them.

Some of the challenges which apply to the use of machine learning approaches are the *class imbalance* problem and the *curse of dimensionality* [40]. These two problems are interrelated for the application of search. Put simply, the class imbalance problem for search is that often there are far more irrelevant than relevant items in a corpus, hampering classification as the classifier can be biased towards the larger of the imbalanced classes. Similarly, when learning from data, such as using visual feature vectors, that as the size of the feature vector grows, an exponential increase occurs in the size of the feature space, correspondingly requiring an increase in the number of examples we require in order to achieve a stable model [41, 42, 43].

[25] explicitly addresses the issue of multi-example, multi-feature search, such as the scenario defined by our example CBMIR system and query, making use of machine learning methods, specifically the use of Support Vector Machines (SVM) [49]. Natsev *et al.* conduct retrieval experiments using the TRECVID 2003 corpus and topics, where each topic

contains not only a statement of the information need but also relevant visual examples. However, as traditional SVMs conduct discriminative classification, they require not only positive training samples (e.g. the visual example queries) but also negative training samples. Natsev *et al.* propose several mechanisms for generating ‘pseudo-negative’ examples, however the best approach was a random sampling of the collection. As TRECVID search topics are typically relatively complex information needs (e.g. find shots of George Bush walking versus a classification task of find outdoor shots), the actual number of relevant shots in a collection is relatively small. Natsev *et al.* refer to this as *bagging*, however it distinguishes itself from traditional machine learning *bagging* approaches as only the negative examples are randomly sampled, whereas in traditional *bagging* approaches, both classes are randomly sampled and multiple classifiers are built and averaged from these random samples. Therefore for each query, a modified bagging approach is conducted, where 10 ‘bags’ are defined, each of which shares the same positive examples, but a random selection of negative examples. This learning process is conducted *per expert*.

[43] also address the issue of imbalanced and sparse learning instances for visual retrieval in multimedia search, by extending the work of [25]. Tešić *et al.* examine the modified bagging approach of [25] offering improvements to both the bagging approach itself and to the selection of the pseudo-negative examples.

### ***Other Methods***

In this section we briefly review other methods which have application for data fusion. We list these approaches under ‘other methods’ as they either occur after the initial retrieval process, or leverage aspects of the underlying data which we cannot for CBMIR.

### ***Relevance Feedback***

Relevance Feedback has long been an area of active research within the information science community [44], dating back nearly 40 years [45]. The idea behind relevance feedback is that it can be difficult for a user to formulate a query to an Information System which will effectively capture what is being sought, but that a user presented with relevant information will recognize it. A system can use these user judgements to better refine the ranking presented to a user, typically in an iterative process. As such it is a valuable technique for bridging the information need of a user to a form that the system can better exploit.

Typically however, relevance feedback is an interactive process, or at the very least occurs after an initial ranking by an information system. Therefore for the purposes of this thesis, we do not consider relevance feedback as one of the approaches which we will examine in depth, as we are concerned with data fusion with regard to the generation of the best initial ranking. Whilst the study of relevance feedback does have some applicability to data fusion, as often multiple forms of evidence need to be combined, we state that the insights we obtain in this thesis can be applied to relevance feedback as a separate

activity in order to improve it, and indeed relevance feedback can be applied to any of our techniques explored here in order to improve retrieval performance.

[44] conduct a thorough review of relevance feedback approaches in image retrieval, whilst [46] in their general review of image retrieval provides some additional updates to this work.

MARS, an early content-based multimedia retrieval system [47], implemented relevance feedback similar to what is found in many text retrieval systems. Using textures as the content-based feature, it uses a vector space representation and investigates both *tf.idf* and Gaussian normalization for determining the weight vector.

Machine Learning approaches feature prominently in image relevance feedback literature, their popularity due in part in being able to explicitly label non-relevant images, so as to achieve a better separation between relevant and non-relevant items [46]. [48] implement relevance feedback using a Support Vector Machine (SVM) [49]. Their paper demonstrates the usefulness of SVMs for the relevance feedback problem, but they highlight the need for multiple positive and negative examples to achieve good accuracy. [54] use an SVM with active learning. Rather than after each round of relevance feedback presenting to the user the top ranked relevant images from a static classifier, their approach is to present images for which the classifier is most uncertain and after each round the classifier determines new decision boundaries. This approach offered improvement over the use of regular SVMs. [50] addresses the class imbalance problem, that is for a typical relevance classification task there are far more non-relevant than relevant images. They build a modified SVM referred to as BSVM (Biased Support Vector Machine) which uses spherical hyperplanes to encompass relevant images. [51] also address the class imbalance problem, through using Semi-Supervised Learning (SSL) and dimensionality reduction in a method they term “Relevance Aggregation Projection (RAP)”. The authors refer to the asymmetry problem in subspace CBIR machine learning, which is that images labeled as ‘relevant’ share some semantic properties, whilst ‘non-relevant’ images have no common properties, only that they differ from the relevant images. By performing dimensionality reduction and SSL they are able to capture nearby unlabeled data points to relevant points, thus leveraging the unlabeled data to improve classification accuracy. For clarity, the difference between active learning and SSL, is that in active learning, the user is required to annotate after each iteration, unlabeled data which a classifier is most unsure of, whilst in SSL, unlabeled data is assigned to relevant and non-relevant labels through transduction. The two approaches can be complimentary.

### ***Query Performance Prediction***

Query Performance Prediction is a related area of work which may have application in the task of weighted data fusion. The objective of query performance prediction is for any given query to a search service, determine how likely it is to provide good results to a user. The typical application for this is to identify queries which may perform poorly and are



therefore good candidates for query re-formulation [52]. The potential application for these approaches is that as they have some discriminating power to determine good from poor performing queries, they may be able to provide a good estimation for values of  $RC_{ij}$ .

## *Conclusions*

In this chapter we conducted an empirical investigation into weighted data fusion and its impact on CBMIR retrieval performance. To achieve our observations we employed a non-conventional experimental model, which directly optimised retrieval performance on test corpora which were being examined. The major benefit of this approach was that for every subsequent experiment that was run within this framework, the ideal set of linear weights for data fusion was always used. This in effect held constant the effect of weighting on retrieval performance and allowed us to make robust observations as to the impact of different factors which effect the data fusion process.

The following observations were made in this chapter:

- The ideal weighting form for data fusion is a highly positively skewed distribution, where specific  $rs_{i,j}$  in a topic are highly weighted, whilst the remaining results receive the remainder of the weight. We found that approximately 10%-20% of the result sets being combined in a query attracted 60%-80% of the weight when ideally weighted.
- Normalisation plays a crucial component in putting evidence into a form which allows it to be easily combined. For score-based normalisation we found that MinMax normalisation produces the best results. For rank-based normalisation we found that our BordaMAX or RankMM approaches produced the best performance. Overall we found that when ideal weighting is used, rank-based normalisation out-performed score-based normalisation.
- We examined the performance of the two most common combination operators, CombSUM and CombMNZ. Whilst CombMNZ is the most popular form of combination operator in text retrieval data fusion literature, we found that CombSUM clearly outperformed CombMNZ.
- Combination levels are often employed in data fusion approaches to make the task of combining evidence from sources more manageable. We found that combining at the ‘expert’ or ‘query’ level produced no difference in retrieval performance. Combining at the ‘direct’ level though produced a very large performance gain over combining results at any arbitrary level, greater than what was anticipated. Typically imposing a combination level restricted retrieval performance to between 50%-75% of what could be achieved with direct combination. This level of performance is better when we consider the first point of this list, that specific pairs correctly weighted drive retrieval performance. The direct level of combination is the only level of combination



Summary of Major Findings	
<i>Ideal Weight Distribution</i>	Highly Positively skewed Log-Normal.
<i>Equivalence Transformations</i>	
Score-based	MinMax
Rank-based	BordaMAX or RankMM
Overall	Rank superior with ideal weights.
<i>Combination Operators</i>	CombSUM outperforms CombMNZ.
<i>Combination Levels</i>	Direct-level far superior to alternatives.
<i>High Impact Queries</i>	Approx. 10% of query-components provide 80% of performance.

Table 1.7: Summary of Major Findings

which allows for the specific weighting of pairs, and therefore is the most effective level of combination.

- We re-examined the early data fusion experiments of Lee, and found that contrary to reported results from text retrieval, when linearly weighted, CombSUM outperforms CombMNZ.

This chapter has explicitly examined many data fusion variables to a very fine level of detail and discovered several properties of data fusion which were masked by sub-optimal weighting being employed. Notable was the assumption that CombMNZ is the ideal form of evidence combination. From these observations, we find that many of the earlier examined algorithms for data fusion do not have properties which would allow for the full exploitation of the ideal data fusion form which we have observed in this chapter. Notably, none of the examined data fusion algorithms allows for the direct weighting of individual pairs  $\langle Expert_i, Query_j \rangle$  at query time, with most data fusion approaches employing a combination level approach which aggregates results at the ‘expert’ level.

This indicates that significant gains can be made through the application of approaches such as saliency analysis to query images, so as to determine if there is a correlation between query images, and those which have been empirically shown to attract the most weight. Development of algorithms which are capable of identifying which pairs  $\langle Expert_i, Query_j \rangle$  are likely to perform well, will generate retrieval performance which surpasses current unsupervised approaches.



## References

- [1] Belkin, N. J., Kantor, P., Fox, E. A., and Shaw, J. A. (1995). Combining the evidence of multiple query representations for information retrieval. *Information Processing and Management*, 31(3):431–448.
- [2] Voorhees, E. M., Gupta, N. K., and Johnson-Laird, B. (1995). Learning collection fusion strategies. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '95)*, pages 172–179, Seattle, Washington, USA.
- [3] Kludas, J., Bruno, E., and Marchand-Maillet, S. (2008). Information Fusion in Multimedia Information Retrieval. *Adaptive Multimedial Retrieval: Retrieval, User, and Semantics: 5th International Workshop, AMR 2007, Paris, France, July 5-6, 2007 Revised Selected Papers*, pages 147–159.
- [4] McGill, M., Koll, M., and Noreault, T. (1979). An evaluation of factors affecting document ranking by information retrieval systems. Technical Report NSF-IST-78-10454 to the National Science Foundation (USA), Syracuse University.
- [5] Das-Gupta, P. and Katzer, J. (1983). A study of the overlap among document representations. *SIGIR Forum*, 17(4):106–114.
- [6] Saracevic, T. and Kantor, P. (1988). A study of information seeking and retrieving, iii: Searchers, searches, overlap. *Journal of the American Society for Information Science and Technology (JASIST)*, 39:177–196.
- [7] Lee, J. H. (1997). Analyses of multiple evidence combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '97)*, pages 267–276, Philadelphia, Pennsylvania, USA.
- [8] Fagin, R. (1996). Combining fuzzy information from multiple systems. In *Proceedings of the Fifteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1996)*, pages 216–226, Montreal, Canada.
- [9] McCabe, M., Chowdhury, A., Grossman, D., and Frieder, O. (2001). System fusion for improving performance in information retrieval systems. In *Proceedings of International Conference on Information Technology: Coding and Computing (ITCC 2001)*, Las Vegas, NV, USA.
- [10] Beitzel, S. M., Jensen, E. C., Chowdhury, A., Grossman, D., Frieder, O., and Goharian, N. (2004). Fusion of effective retrieval strategies in the same information retrieval system. *Journal of the American Society for Information Science and Technology*, 55(10):859–868.

- [11] Snoek, C. G. M., Worring, M., and Smeulders, A. W. M. (2005). Early versus Late Fusion in Semantic Video Analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia (MULTIMEDIA '05)*, pages 399–402, Singapore, Singapore.
- [12] Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A., and Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380.
- [13] Van Rijsbergen, C. (1979). *Information Retrieval, 2nd edition*. Butterworth-Heinemann Newton, MA, USA.
- [14] Blanken, H. M., Vries, A. P. d., Blok, H. E., and Feng, L. (2007). *Multimedia Retrieval (Data-Centric Systems and Applications)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- [15] Manjunath, B., Salembier, P., and Sikora, T., editors (2002). *Introduction to MPEG-7: Multimedia Content Description Language*. Wiley.
- [16] Metzler, D. and Croft, W. B. (2007). Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274.
- [17] Cooke, E., Ferguson, P., Gaughan, G., Gurrin, C., Jones, G., le Borgne, H., Lee, H., Marlow, S., Donald, K. M., McHugh, M., Murphy, N., O'Connor, N., O'Hare, N., Rothwell, S., Smeaton, A. F., and Wilkins, P. (2004). TRECVID 2004 Experiments in Dublin City University. In *Proceedings of TRECVID 2004*, Gaithersburg, MD, USA.
- [18] Amir, A., O Argillander, J., Berg, M., Chang, S.-F., Franz, M., Hsu, W., Iyengar, G., R Kender, J., Kennedy, L., Lin, C.-Y., Naphade, M., (Paul) Natsev, A., Smith, J. R., Tesic, J., Wu, G., Yan, R., and Zhang, D. (2004). IBM Research TRECVID-2004 Video Retrieval System. In *Proceedings of TRECVID 2004*, Gaithersburg, MD, USA.
- [19] Jeong, S., Kim, K., Chun, B., Lee, J., and Bae, Y. J. (1999). An effective method for combining multiple features of image retrieval. In *TENCON 99. Proceedings of the IEEE Region 10 Conference*, volume 2, pages 982–985, Silla Cheju, Cheju Island, Korea.
- [20] Jin, X. and French, J. C. (2003). Improving image retrieval effectiveness via multiple queries. In *Proceedings of the 1st ACM international workshop on Multimedia databases (MMDB '03)*, pages 86–93, New Orleans, LA, USA.
- [21] MPEG-7 (2001). Multimedia Content Description Interface. Standard No. ISO/IEC n=15938.

- [22] Smeaton, A. F., Over, P., and Kraaij, W. (2006). Evaluation Campaigns and TRECVID. In *Proceedings of the 8th ACM SIGMM International Workshop on Multimedia information retrieval (MIR 2006)*.
- [23] Clough, P., Grubinger, M., Hanbury, A., and Müller, H. (2008). Overview of the imageclef 2007 photographic retrieval task. In *Proceedings of the CLEF 2007 Workshop*, LNCS, Budapest, Hungary.
- [24] Yan, R. and Hauptmann, A. G. (2006). Probabilistic latent query analysis for combining multiple retrieval sources. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*, pages 324–331, Seattle, Washington, USA.
- [25] Natsev, A. P., Naphade, M. R., and Tesic, J. (2005). Learning the semantics of multimedia queries and concepts from a small number of examples. In *Proceedings of the 13th annual ACM international conference on Multimedia (MULTIMEDIA '05)*, pages 598–607, Singapore, Singapore.
- [26] McClave, J. T. and Sincich, T. (2006). *Statistics (10th Edition)*. Prentice Hall.
- [27] Renda, M. E. and Straccia, U. (2003). Web metasearch: rank vs. score-based rank aggregation methods. In *Proceedings of the 2003 ACM symposium on Applied computing (SAC '03)*, pages 841–846, Melbourne, FL, USA.
- [28] Montague, M. and Aslam, J. A. (2001). Relevance score normalization for metasearch. In *Proceedings of the 10th international conference on Information and knowledge management (CIKM '01)*, pages 427–433, Atlanta, Georgia, USA.
- [29] Dwork, C., Kumar, R., Naor, M., and Sivakumar, D. (2001). Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web (WWW10)*, pages 613–622, Hong Kong, China.
- [30] Fox, E. A. and Shaw, J. A. (1994). Combination of Multiple Searches. In *Proceedings of the 3rd Text REtrieval Conference (TREC-2)*, Gaithersburg, MD, USA.
- [31] McDonald, K. and Smeaton, A. F. (2005). A comparison of score, rank and probability-based fusion methods for video shot retrieval. In *Proceedings of the 4th ACM international Conference on Image and Video Retrieval (CIVR '05)*, Dublin, Ireland.
- [32] Croft, W. B. (2000). Combining approaches to information retrieval. *Advances in Information Retrieval*, pages 1–36.
- [33] Robertson, S. (2007). On score distributions and relevance. In *29th European Conference on Information Retrieval (ECIR 2007)*, pages 40–51, Rome, Italy.

- [34] Vogt, C. C. and Cottrell, G. W. (1999). Fusion Via a Linear Combination of Scores. *Information Retrieval*, 1(3):151–173.
- [35] Yan, R. and Hauptmann, A. G. (2003). The combination limit in multimedia retrieval. In *Proceedings of the eleventh ACM international conference on Multimedia (MULTIMEDIA '03)*, pages 339–342, Berkeley, CA, USA.
- [36] Yan, R., Yang, J., and Hauptmann, A. G. (2004). Learning query-class dependent weights in automatic video retrieval. In *Proceedings of the 12th annual ACM international conference on Multimedia (MULTIMEDIA '04)*, pages 548–555, New York, NY, USA.
- [37] Chua, T.-S., Neo, S.-Y., Li, K.-Y., Wang, G., Shi, R., Zhao, M., and Xu, H. (2004). TRECVID 2004 search and feature extraction task by NUS PRIS. In *Proceedings of TRECVID 2004*, Gaithersburg, MD, USA.
- [38] Kennedy, L. S., Natsev, A. P., and Chang, S.-F. (2005). Automatic discovery of query-class-dependent models for multimodal search. In *Proceedings of the 13th annual ACM international conference on Multimedia (MULTIMEDIA '05)*, pages 882–891, Singapore, Singapore.
- [39] Whitten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques; (2nd ed.)*. Morgan Kaufmann Publishers, San Francisco, CA, USA.
- [40] Akbani, R., Kwek, S., and Japkowicz, N. (2004). Applying Support Vector Machines to Imbalanced Datasets. In *Proceedings of the 15th European Conference on Machine Learning (ECML 2004)*, pages 39–50, Pisa, Italy.
- [41] Bellman, R. (1961). *Adaptive control processes: a guided tour*. Princeton University Press, Princeton, N.J., USA.
- [42] Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is nearest neighbor meaningful. In *7th International Conference on Database Theory (ICDT 1999)*, pages 217–235, Jerusalem, Israel.
- [43] Tešić, J., Natsev, A., Xie, L., and Smith, J. (2007a). Data modeling strategies for imbalanced learning in visual search. In *IEEE International Conference on Multimedia and Expo (ICME 2007)*, pages 1990–1993, Hannover, Germany.
- [44] Zhou, X. S. and Huang, T. S. (2003). Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8(6):536–544.
- [45] Rocchio, J. J. (1971). Relevance feedback in information retrieval. In Salton, G., editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, Prentice-Hall Series in Automatic Computation, chapter 14, pages 313–323. Prentice-Hall, Englewood Cliffs NJ.

- [46] Datta, R., Joshi, D., Li, J., and Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60.
- [47] Rui, Y., Huang, T., and Mehrotra, S. (1997). Content-based image retrieval with relevance feedback in MARS. *Image Processing, 1997. Proceedings., International Conference on*, 2:815–818 vol.2.
- [48] Hong, P., Tian, Q., and Huang, T. S. (2000). Incorporate support vector machines to content-based image retrieval with relevant feedback. In *International Conference on Image Processing (ICIP 2000)*, Vancouver, BC, Canada.
- [49] Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.
- [50] Hoi, C.-H., Chan, C.-H., Huang, K., Lyu, M., and King, I. (2004). Biased support vector machine for relevance feedback in image retrieval. *Proceedings of the 2004 IEEE International Symposium on Neural Networks (ISSN2004)*, 4:3189–3194.
- [51] Liu, W., Jiang, W., and Chang, S.-F. (2008a). Relevance aggregation projections for image retrieval. In *Proceedings of the 7th ACM international Conference on Image and Video Retrieval (CIVR '08)*, pages 119–126, Niagara Falls, Canada.
- [52] Cronen-Townsend, S., Zhou, Y., and Croft, W. B. (2002). Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 299–306, Tampere, Finland.
- [53] Xie, L., Natsev, A., and Tesic, J. (2007). Dynamic Multimodal Fusion in Video Search. In *IEEE International Conference on Multimedia and Expo (ICME 2007)*, pages 1499–1502, Hannover, Germany.
- [54] Tong, S. and Chang, E. (2001). Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia (MULTIMEDIA '01)*, pages 107–118, Ottawa, Canada.