A MODEL FOR MULTIMODAL INFORMATION RETRIEVAL

Rohini K. Srihari, Aibing Rao, Benjamin Han, Srikanth Munirathnam, Xiaoyun Wu

{rohini | arao | dh6 | srikanth | xwu}@cedar.buffalo.edu

CEDAR
Suite 202, UB Commons, State University of New York at Buffalo
Amherst, New York 14228-2567, USA

ABSTRACT

Finding useful information from large multimodal document collections such as the WWW without encountering numerous false positives poses a challenge to multimodal information retrieval systems (MMIR). A general model for multimodal information retrieval is proposed by which a user's information need is expressed through composite, multimodal queries, and the most appropriate weighted combination of indexing techniques is determined by a machine learning approach in order to best satisfy the information need. The focus is on improving *precision* and *recall* in a MMIR system by optimally combining text and image similarity. Experiments are presented which demonstrate the utility of individual indexing systems in improving overall average precision.

Keywords: Multimodal Information Retrieval, Content-based Retrieval, Image Retrieval, Multimodal Query, Machine Learning

1. INTRODUCTION

Multimodal Information Retrieval presents a great challenge since it deals with several data types/modalities, each having its own retrieval models. The body of literature in multimodal IR is vast, ranging from logic formalisms for expressing the syntax and semantics of multimodal queries [2] to MPEG-4 [3] standards for video coding which calls for explicit encoding of semantic scene contents. A popular approach has been to add a layer representing *meta querying* on top of the individual retrieval models. An agent-based architecture for decomposing and processing multimodal queries is discussed in [1]. In focusing so much on formalisms, especially in the logic-based approaches, researchers sometimes make unrealistic assumptions about the quality of information that can be automatically extracted (e.g., the detection of complex temporal events in video). HM-SQL[6], a query language for retrieving hybrid, multimodal data, attempts to extend and enhance SQL to support querying of multimodal entities. However, it suffers from the inevitable manual preparation of the semantics content, and unrealistically assumes that for each media source its specialized index and query language, acting as a black box, suffices to retrieve semantically relevant information. The black box approach thus bars the important synergy between indexing techniques across modalities.

This paper is about image retrieval using techniques for multimodal IR; the focus is on utilizing *automatically* extracted information from multimodal data to improve retrieval. For example, in order to retrieve pictures of a particular person, a judicious combination of techniques such as face detection, recognition, specialized similarity matching, as well as text indexing may be required. Furthermore, special operators for combining the various results are required.

Ideally for a multimodal IR model to be successful, a composite multimodal query must be first decomposed into the individual query components, i.e., the semantics of the query must be determined accurately. The decomposition makes it possible to combine multiple lower-level indexing schemes - the individual rankings obtained from the indices are then combined to maximize the expected precision and recall over all possible combination schemes. Relevance feedback mechanisms can thus be introduced to each index through an intuitive and effective user interface.

One particular technique for relevance feedback can be readily applied in a multimodal framework. The basic idea is that if a certain indexing function has a high expected utility, it may contribute to a modification of a query component to be used in another indexing function. For example, if image similarity has a high expected utility, the text query string can be modified to be similar to the text components of top-ranked documents based on image similarity.

2. SEMANTICS OF MULTIMODAL QUERIES: INFORMATION NEED

When a user presents a query, either as a text string, or as a composite query involving several modalities, she is attempting to express her *information need*. Information need is an abstract concept: it represents everything she is looking for, and no more. In the case of multimodal especially, a user typically recognizes when a document matches her information need, but may not be able to describe it in words. Information need is approximated through *concrete* queries. These queries can consist of a single modality, as with text search, or consist of several modalities, referred to as a *composite query*. Queries can either represent a subset of information needs, or serve as an example of a document matching the information need. In many cases, a composite query is required in order to express an information need. An example of this is *find pictures like this that were taken in the summer of 1998*. Without some external source of context, this query cannot be satisfied.

Obviously, certain modalities are more suited for expressing an information need. For example, an image query may be better suited for finding pictures than a text description, if the artistic qualities of the picture are the central focus. An interesting case is where a composite query is used when one is not required. That is, *redundancy* is present in the composite query. Although a picture query matches the information need best, a redundant text query may also be included. Such queries represent a realization of the weakness of indexing methods associated with a certain modality. If image matching were perfect, then a redundant text query would not be necessary.

Fig. 1 illustrates the three levels of multimodal retrieval. At the highest level, there is the abstract information need. The user approximates her information need through queries, single or composite; these are referred to as *modal query objects*. Each of these are in turn processed through the *query decomposition* phase into *query components*. Examples of query components that are to be matched include **text strings**, **image features** such as color histograms, **object categories** (e.g., person, Bill Clinton, car), **spatial relationships** (e.g. person on top of car), **picture attributes** such as indoor/outdoor, as well as **metadata** such as date, time, etc. Clearly, the number and types of query components will increase as new media sources such as audio and video are introduced.

At the last stage, query components are matched using a judicious combination of the low-level indexing techniques.

Let $\mathcal{D} = \{D_1, D_2, ..., D_m\}$ be the set of all data items. Each low-level indexing function $indx_i$ ranks the document database based on certain criteria.

Given an indexing function $indx_i$ and a query component q_j, let

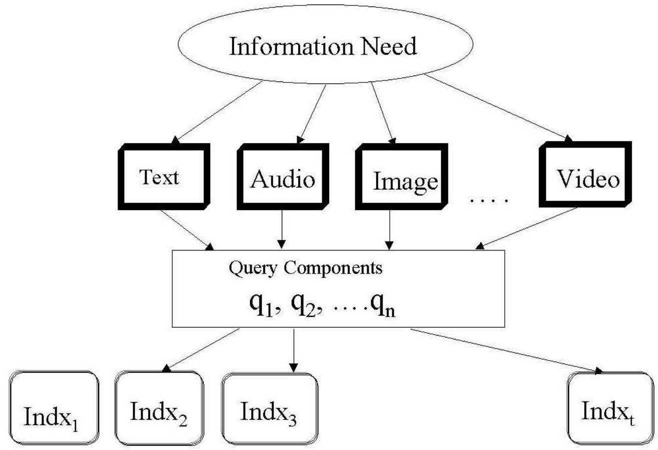

Figure 1: Translating multimodal queries into low-level indexing techniques

$$sim_i^j : \mathcal{D} \longrightarrow I$$

be the normalized similarity between q_j and the j^{th} component of each document with respect to $indx_i$; I is the closed interval of real numbers from 0.0 to 1.0. Sort $\mathcal{D}$ using $sim_i^j(D_k)$ as the key of document D_k. This results in a ranking of the database, $\mathcal{D}_i^j$, which is the match for the query based on the j^{th} component under indexing scheme ind_i. The ranking is a permutation of $\mathcal{D}$. Since each of these similarity functions have inherent weaknesses, one interesting issue is how to properly combine these functions to best satisfy the user's information need.

3. MAXIMIZING EXPECTED UTILITY

This section discusses a model whereby the process of satisfying the user's information need may be viewed as maximizing the expected utility of the various similarity functions. It involves a training phase for determining the suitability of a given similarity function sim_i^j.

Let $\mathcal{P}$ be the set of all permutations of the data set $\mathcal{D}$. Each element of $\mathcal{P}$ may be regarded as a ranking of the database representing a match to some given query. Let $\mathcal{D}_{opt} \in \mathcal{P}$ be an optimal ranking in that it has the best average precision for that query. The average precision of the retrieval in response to a query is defined as

$$\mathbf{AP} = \frac{\mathbf{1}}{\mathbf{R}} \sum_{\mathbf{i=1}}^{\mathbf{R}} \frac{\mathbf{i}}{\mathbf{N_i}}$$

where R is the total number of relevant documents and N_i is the number of documents retrieved at the time when the i^{th} relevent document is retrieved. This serves conveniently as our one-number performance metric for a particular combination of query components and indexing schemes. Based on the ground truth provided by $\mathcal{D}_{opt}$, a performance utility with respect to the query can be defined as:

$$\mathbf{AP} : \mathcal{P} \longrightarrow \mathcal{R}^+$$

which maps each permutation to the average precision of the permutation, here $\mathcal{R}^+$ is the set of all non-negative reals. Average precision combines precision and recall in one measure; this is the quantity to be maximized in the model being described.

The goal is to approximate $\mathcal{D}_{opt}$, that is to maximize the average precision of the original query based on the performance of the individual component matches. The goal of the multimodal IR system is to select the weights ω_i^j that maximize the average precision of the permutation $\mathcal{D}_c \in \mathcal{P}$:

$$\mathcal{D}_c = P(\hat{\omega}_i^j)$$

which is generated by sorting $\mathcal{D}$ with key computed by applying $\sum_{i,j} \hat{\omega}_i^j sim_i^j$ to each data item. This represents the most typical case, *linear combination*. The combination can also be a non-linear function.

A training phase is now employed in order to determine the weights ω_i^j. The training data consists of a set of queries $Q_1, Q_2, ..., Q_n$ as well as the corresponding relevance judgement. For each Q_k, decompose it into the corresponding set of query components q_j. For a weight ω_i^j, (i) apply $\sum_{i,j} \omega_i^j sim_i^j$ to each data item (ii) sort $\mathcal{D}$ using the above values as a sorting key, resulting in the permutation $P_k(\omega_i^j)$, and (iii) then calculate $\mathbf{AP}(P_k(\omega_i^j))$. Finally, the average(AAP)

$$AAP = \frac{1}{n} \sum_{k=1}^{n} \mathbf{AP}(P_k(\omega_i^j))$$

is the overall performance by employing weights ω_i^j for the set of test queries. The goal is to maximize this performance which corresponds to determining the optimal set of weights $\hat{\omega}_i^j$.

At the retrieval stage, a query is decomposed in a similar fashion. The component queries q_j are used to compute sim_i^j; then sort $\mathcal{D}$ using $\sum_{i,j} \hat{\omega}_i^j sim_i^j$ values as sort keys.

The above training phase is for finding a single set of optimal weights so that a query can be satisfied optimally *on average* by adopting the weights. However, this may not give the best performance for a particular query. This problem can be alleviated by preparing a set of most representative queries beforehand, and then computing an optimal set of weights for each of the representatives. Intuitively the representative queries *evenly* cluster the entire database, and each subsequent new query should fall in one of these clusters. The set of weights for combining indices is then picked such that the corresponding representative query is *closest* to the new query. Formally speaking, let $Q_1, Q_2, ..., Q_N$ be the representative queries. For each Q_k, we then find the optimal weights $\{\omega_i^j\}_k$. In the retrieval phase, for any query Q we first compute the similarity distance between Q and each Q_k, denoted as $d(Q, Q_k)$, then $\{\omega_i^j\}_{k_0}$ is the weights we use for indices combination such that

$$k_0 = \arg \min_{1 \le k \le N} d(Q, Q_k)$$

Finally, the training process can be done on several different models combining various indexing techniques, using for example linear combination or filtering. The main drawback of the linear combination is that all of the the indexing operations need to be performed before the combination. But in some situations, one indexing operation may be more costly than another, so that we may perfer to apply it to the filtered results of the other one. As an example, consider the query *find pictures of Bill Clinton during the last election*. For efficient retrieval the captions of the pictures are first used to filter out potentially irrelevant images, then the face detection algorithm is applied to give the final ranking list, since the latter operation is more computationally intensive. The command to perform this is:

```
Filter((SIM(Text, q_{text})),
       SIM(attr=faces))
```

This gives rise to the *optimal sequence problem*: Which sequence of indexing operations will most efficiently yield the best retrieval performance? Thus in general the indexing operations in a sequence are performed one by one instead of being performed together as in the linear combination model. This problem is not investigated in the paper.

4. EXPERIMENTS AND RESULTS

The dataset contains nearly 2,500 documents (images with accompanying text) gathered from various websites according to 50 selected topic areas. Two queries are formulated for each topic area for which 10 relevant documents and 40 non-relevant documents exist. In total, 100 queries are formulated with well-defined relevance judgments for the dataset. Half of these queries will be used for training while the rest will be used for testing. To demonstrate the retrieval improvement achieved by combining image and text similarities in the model, experiments are performed on these 50 Topic Databases (TDB) and on the Combined Database (CDB) with two training strategies described in section 3: (i) Find the weights which give the optimal average performance (T_1); (ii) For each query in the training data, find the weights which give optimal performance on the query (T_2). The image similarity is calculated based on the triangular geometric color histogram [4] and the collateral text similarity is calculated using SMART [5]. Based on the weights obtained from both training techniques, the performance of 50 queries in the testing data are calculated and compared.

With technique T_1, the optimal image weights learned with the training queries are 0.4 for TDB and 0.7 for CDB. With technique T_2, part of the optimal image weights corresponding to each of the training queries are enumerated in Fig. 2 for both databases.

Query #	1	2	3	4	5	6	7	...
TDB	0.5	0.7	0.5	0.1	0.7	0.3	0.5	...
CDB	0.7	0.6	0.5	0.8	0.8	0.4	0.6	...

Figure 2: The optimal image weights for each training query by applying T_2 on TDB and CDB. They are obtained by optimizing its average pricision in linear combination model.

Now use the weights obtained with T_1 and T_2 to test the performance of the 50 test queries on each database. In the case of $T2$, for each of the queries, find the most similar query in the training set and use the image weight corresponding to the training query in Fig. 2 to do a linear combination. The average precisions are ploted in Fig. 3 which shows that training strategy T_2 is much better than T_1 for both TDB and CDB.

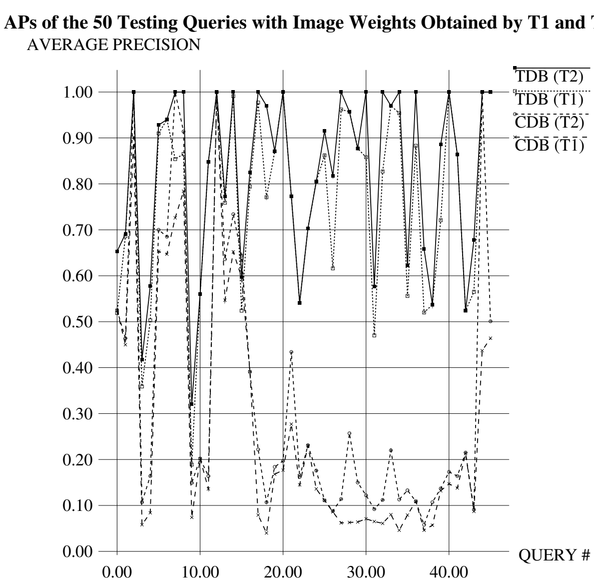

Figure 3: The average precisions(AP) of the testing query with the weight obtained with techniqe T_1 and T_2. The training strategy T_2 performs better than T_1 for both TDB and CDB.

To summarize, the average values of the APs over the 50 testing queries for both strategies are shown in Fig. 4, which also shows the retrieval performance when using single modality (image similarity or text indexing) for comparison.

DB	*Text Sim.*	*LC(T1)*	*LC(T2)*	*Image Sim.*
TDB	0.693	0.767	**0.819**	0.487
CDB	0.204	0.267	**0.332**	0.147

Figure 4: The average values of the APs over the 50 testing queries for different retrieval techniques.

5. CONCLUSION

It is necessary to combine indexing and retrieval techniques for different modality effectively to perform multimodal information retrieval. For such a combination to be successful, a machine learning and relevance feedback technique is needed. The learning techniques discussed in this paper take advantage of the simplicity of a linear combination, but still have an inevitable drawback: it is difficult to set up the training queries. For techniqe T_2, the problem of setting up the training queries for a given database is essentially equivalent to the problem of clustering the database.

6. REFERENCES

[1] M. B. and F. Dubois. An Agent-based Architecture for Content-Based Multimedia Browsing. In M. T. Maybury, editor, *Intelligent Multimedia Information Retrieval*. AAAI Press/MIT Press, 1997.

[2] C. Meghini. An Image Retrieval Model Based on Classical Logic. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 300–309. ACM, 1995.

[3] What is mpeg-4. http://www.crs4.it/˜luigi/MPEG/mpeg4.html, 1998.

[4] A. Rao, R. K. Srihari, and Z. Zhang. Geometric histogram: A distribution of geometric configurations of color subsets. To appear in *Proceedings of SPIE: Internet Imaging*, 3964, January 2000.

[5] G. Salton. *Automatic Text Processing*. Addison-Wesley, 1989.

[6] V. S. Subrahmanian. *Priciples of Multimedia Database Systems*. Morgan Kaufmann, 1998.