Building Up of Image and Multimedia Object Index Through Continuous Usage

C. H. C. Leung, M. Q. Luo
Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong.
clement@comp.hkbu.edu.hk

Abstract—Unlike text documents, a key problem facing the search of images and multimedia data objects is the ability to index them properly. Here, we describe a mechanism for indexing through usage, which makes use of a dynamic evolutionary approach. By capturing, analyzing and interpreting user response and query behavior, the patterns of searching and finding multimedia data objects may be established. Within the present architectural paradigm, the semantic index may be dynamically constructed, validated, and built-up, where the performance of the system will increase as time progresses. Our system also incorporates a high degree of robustness and fault-tolerance whereby inappropriate index terms will be gradually eliminated from the index, while appropriate ones will be reinforced. We also incorporate genetic variations into the design to allow objects which may otherwise be hidden to be discovered. Experimental results indicate that the present approach is able to confer significant performance benefits in the semantic searching and discovery of a wide variety of multimedia data objects, such as images, music, and videos.

Keywords: Concept-based, Dynamic Indexing, Genetic Algorithms, Index Hierarchy, Multimedia Information Search, Relevance Feedback and Semantics.

1 Introduction

The PageRank algorithm for ranking query results makes use of scoring and the link structure of the Web [3], [5], [6]. By applying the PageRank algorithm, the relative importance of hyperlinked set of documents can be measured. However, multimedia information search is far more difficult than searching text-based documents since the content of text-based documents can be extracted automatically and relatively easily while the content of multimedia objects cannot be done in a similar way.

Various methods and techniques have been proposed for retrieving images. They are mainly classified into two main categories; “concept-based” image retrieval, and “content-based” image retrieval [1], [2], [9], [11], [12]. The former focuses on using words to retrieve images (e.g. title, keywords, and caption), while the latter focuses on the visual features of the image (e.g. size, colours, and textures). In an effective “concept-based” multimedia retrieval system, efficient and meaningful indexing is necessary [4]. Due to the current technology limitations, it is impossible to extract semantic content of the multimedia objects automatically [13]. Meanwhile, the discovery and insertion of new indexing terms is always costly and time-consuming. Therefore, some manual indexing is required for an initial multimedia searching system.

In addition, Web 2.0 relies heavily on user-generated content and users’ expert knowledge [15]. The Wikipedia is one of the successful examples of the web 2.0. In our proposed method, we adopted the spirit of the web 2.0 and proposed the collaborative indexing by the Internet users. Through the continuous and extensive use of the multimedia search system, users tend to provide more meaningful indexing and expert judgment for the system. Consequently, multimedia objects would be optimally indexed such that semantic visual information search on those objects would become possible.

2 System Organisation

The structure of our system consists of different levels for supporting the multimedia search. It includes user level, interface level, and database level. The database level contains two sublevels, query level and index level. The organization of the system is shown in Figure 1.

User and System Interface Level
User is the one who interact with the search system via the system interface. They can search multimedia objects by submitting query with keywords. After the system processes is done, they can get search results from the system interface and provide feedback through the system interface. The system
A system interface acts as a bridge between user and the system database. It captures user query input with search criteria, search result feedback and selection from the user. Also, it presents system search result to the user.

**Index Level**
Index level consists of the index hierarchy and index table. It partitions index levels with parameters values on index scores. Index terms are linked with related multimedia objects with its corresponding index score. Multimedia objects are indexed with different index terms with its corresponding score.

**Fig. 1** System Organisation

**Database Level**
The system database level is the core part of the system. It consists of a query level and an index level. The query level consists of stored procedures that are responsible for all database related processes, such as index insertion, index score updating, object ranking and retrieval, for the entire system. The index level is an index table, with an index hierarchy, which linking multimedia objects and index terms with the corresponding index score. The index score means the importance of the index term that related to the multimedia object. The following sections will focus on discussing the system database level.

### 3 Index Level

In our study, we only focused on the indexing of semantic contents of multimedia objects and exclude the metadata [9]. Since the indexing of metadata is relatively straightforward and less meaningful than the semantic contents that perceived by human. For example, indexing and retrieving a song by its characteristics (e.g. style) is more meaningful than indexing its metadata (e.g. track number).

**Index Element and Multimedia Object**
We considered a set of data objects \( \{O_j\} \). In addition, the characteristics or semantic contents of each element object \( O_j \) (multimedia data objects such as images, video, or music) in this set can not be extracted automatically. For every \( O_j \), it links with a set of index \( I_j \) that consists of a number of elements:

\[
I_j = \{e_{j1}, e_{j2}, \ldots, e_{jM_j}\}.
\]

Each index element \( e \) is a triple, such that

\[
e_{jk} = (t_{jk}, s_{jk}, o_j),
\]

where \( t_{jk} \) is an index term ID, \( s_{jk} \) is the score associated with \( t_{jk} \), and \( O_j \) is the object ID. The higher the score \( s_{jk} \), the index term \( t_{jk} \) is more important to the object \( O_j \). The relationship of \( t_{jk}, s_{jk}, O_j \) can be represented in the following entities.

- **IndexTerm** (index_id, index_term)
- **MultimediaObject** (object_id, object_name, object_description, …)
- **IndexTable** (index_id, object_id, score, …)

Each index term is uniquely identified by the primary key, index_id. Similarly, each multimedia is uniquely identified by the primary key, object_id. In the index table, each item is uniquely identified by the composite key, index_id and object_id.
Index Hierarchy

The index hierarchy refers to the index sets of all the objects stored in the database. By partitioning the value of score $s_{jk}$, it can be divided into N levels $L_1, L_2, ..., L_N$ with using a set of parameters $P_1, P_2, ..., P_N$. For example, $P_1=0$, $P_2=10$, $P_3=20$, and $P_4=30$. There would be 4 levels of the index set. (See Fig. 2.)

![Fig. 2. Partitioning of the Index Set](image)

By considering score value $x$ of a given index term, the index term would be placed in level $L_i$ if

$$P_i < x < P_{i+1},$$  \hspace{1cm} (3)

and would be placed in level N if

$$P_i \leq x.$$  \hspace{1cm} (4)

In this index hierarchy, the higher the level, the more significant than those in the lower levels to an object. Hence, multimedia data would be searched in the top level first. In some case, we may neglect the lower levels.

Minimal Index

When an object $O$ is minimally indexed, it means:

(i) $O$ has only a single index term $T$, and
(ii) $T$ is a single word.

Since the multimedia objects are searched by search term(s) in a search query, the objects should be indexed in order to be searched. Therefore, it is necessary to add a minimal index to a multimedia object when user adding objects to the database. The unindexed multimedia object, which is not indexed by any index term, could not be found.

4 Query Level

Stored procedures in the query level are responsible for all database related processes for the entire system. Those processes include multimedia object retrieval, search result ranking, index score updating, index insertion, etc.

Index Score and Object Ranking

When user input a series of search terms $T_1, T_2, ..., T_n$ in a search query $Q(T_1, T_2, ..., T_n)$, the query score $S(Q|O_j)$ for an object $O_j$ can be extracted by the following SQL statement:

```sql
SELECT score
FROM IndexTable
WHERE object_id = O_j-ID
and index_id = T_i-ID;
```

This score implies the relative importance of an index term $T_i$ to the corresponding object $O_j$. Thus, the multimedia objects in the query result should be ordered by score in descending order. The query result can be obtained by the following SQL statement:

```sql
SELECT object_id, SUM(score) AS s
FROM IndexTable
WHERE index_id in (T_1-ID, T_2-ID, ..., T_n-ID)
GROUP BY object_id
ORDER BY s DESC
```

5 Term Scoring Algorithms

User search behaviors, such as result selection and the relevance feedback, would affect the index score directly. By continuously use of the search system, more user search behaviors can be collected and analyzed. The following will introduce how the scores are affected by the user search behaviors.

In this section, we will consider an example on a user input search query $Q(T_1, T_2,)$, N multimedia objects $O_1, O_2, ..., O_n$ are returned in the query result and ordered by the corresponding score $S_1, S_2, ..., S_n$ in descending order.

Score Increment

By considering the example, the related index scores on $T_1$ and $T_2$ for the desired object $O_x$ would be increased by the following cases:

(i) When user select $O_x$ in the query result list, or
(ii) When user provide positive feedback on $O_x$

In these two cases, the related index scores on $T_1$ and $T_2$ for the desired object $O_x$ would be increased by a predefined value $\Delta_x$, where the predefined value for these two cases can be different.

Score Decrement

By considering this example, there are two cases that would cause the index score decrease:

(i) When user provide negative feedback on $O_x$, the related index scores on $T_1$ and $T_2$ for the desired object $O_x$ would be decreased, or
(ii) When user do not click on any object on the query result list, the related index scores on \( T_1 \) and \( T_2 \) for all objects \( O_1, O_2, \ldots, O_n \) in the query result would be decreased.

In these two cases, the score would be decreased by a predefined value \( \Delta_y \), where the predefined value for these two cases can be different.

6 Growth of Index

As a review of the index growth approach, consider an object \( K \) that is minimal indexed with a term \( T_1 \). \( K \) can be searched by user query which contains \( T_1 \). It may have chance that many objects returned in the query result since many objects are indexed with \( T_1 \). Among these returned objects, user can distinguish objects by adding another index term \( T_2 \) to \( K \). Thus, user can search the desired object by entering both index terms in the search query [9].

For example, when we consider the searching of a song “Für Elise”, which is a piece of music composed by Beethoven. Initially, we assume that the audio object is minimally indexed with the term “Beethoven”. User can search this song by the term “Beethoven”. Sometimes, some user query would be more specific, with both search terms “Beethoven” and “Für Elise” are used. Same multimedia object would be returned in the result when searching by the term “Beethoven”, since the term “Für Elise” is not indexed yet. Eventually, user would select the audio object “Für Elise” and suggest a new index term, “Für Elise”, to this music. Thus, the new index term would be included in the low level of the index hierarchy for this audio object. For every query that having both terms, “Beethoven” and “Für Elise”, user would select this audio object and increase the score on the index terms for that object. Thus, the score of the index would be increased and the new index would be proper installed. (The increase of score will be introduced in the next section.) Through progressive usage, the indexing on multimedia objects would be enriched. This example is explained in Fig. 3.

![Fig. 3. An Example on Index Growth](image-url)
7 Query Results Optimization Based on Genetic Algorithms

The use of Genetic Algorithms (GA) has been widespread and this has been applied in search engines in the optimizing of results [10]. After massive use of the system, the high scored multimedia objects always ranked very top on the search results. Generally, users tend to be interested in the high ranked objects. Consequently, the higher scored objects always have higher chance of increasing score while the chance for the new added objects appearing in the result list would be reduced. Thus, the new added objects would be ranked very low and nearly “hidden”.

In order to optimize the search results, our search system should provide variations in the result. When we consider a huge number (e.g. 1000+) of multimedia objects returned by a query result, user may not reach their desired object since the object always ranked very low and nearly “hidden”. With the randomness characteristics of Genetic Algorithm (GA), those “hidden” objects would promote to a higher ranking position and discovered eventually.

We proposed to apply the GA in discovering those “hidden” objects. With the GA, the object ranking for each object in the query result is determined by a probability value:

\[ P_i = \frac{S_i}{\sum_{j=1}^{n} S_j} \]

where \( i = 1, 2, \ldots, n \).

(5)

The higher the probability value would have greater chance of getting higher rank in the query result list.

8 Tagging vs. Indexing

Indexing is widely applied in today’s search engines, while tagging is also commonly used in Web 2.0 searching websites, such as Flickr, Gmail, YouTube, etc. In this study, we will compare these two approaches and distinguish their similarities and dissimilarities.

An index is a systematic hierarchy that enables fast and accurate data retrieval. A tag is a term or keyword associated with a piece of information, such as a document, a picture, a video, etc. Indexing and tagging allows users to search information by entering indexed / tagged words into query. Both indexing and tagging relies on the “words” that indexed / tagged on the information, however, the content of the “words” would be different. In indexing approach, the index term can be a related keyword, or a term, or a phrase that can have semantic meanings. Stop-words, such as a, an, the, etc., would not be indexed. [7], [8] But, in tagging approach, most of the tags, usually are uncontrolled keywords or terms, are nouns and without semantics.
By comparing their structure, indexing maintains a more well-defined hierarchy levels structure while tagging has no hierarchy and tags are unconnected. [14] As for the installation of these approaches, tagging only allows input tags manually while indexing can be done by search engine automatically and input manually.

9 Experiments

To examine the possibility of the system, we carried out experiments on showing the growth of index and how multimedia objects can be found after indexing.

Index Growth
Initially, 100 of multimedia objects are stored in the system database. Each object is initially indexed with an index term. The index table is grown through 60 queries, inputted by user. Throughout the queries, user randomly add new index term(s) to the query results. Fig. 4 shows the relationship between number of queries and the number of index in the system.

Comparison Between Before and After Indexing
Before indexing through user queries, multimedia objects can be only found by its initial index. However, with the growth of the index table, multimedia objects can be found by other index term(s) that added by users. Table 1 and 2 show the query details and its results (before and after indexing). Also, the recall-precision graph for query results after indexing is shown in Fig. 5. In this experiment, user used new index terms as the input queries. Before indexing, null of the items is retrieved since the multimedia objects are not indexed with new index terms. After indexing through 60 queries, multimedia object(s) can be found by using the same query.

<table>
<thead>
<tr>
<th>Targeted Multimedia Object ID</th>
<th>User Query</th>
<th>Query Result</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>concerto</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>3</td>
<td>blue, danube</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>4</td>
<td>flight, bee</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>8</td>
<td>concerto</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>8</td>
<td>piano</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>38</td>
<td>handel, piano</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>58</td>
<td>nature</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>58</td>
<td>boy</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>71</td>
<td>think</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
<tr>
<td>71</td>
<td>me</td>
<td>not found</td>
<td>0</td>
<td>undefined</td>
</tr>
</tbody>
</table>

Table 1. Query Results Before Indexing

<table>
<thead>
<tr>
<th>Targeted Multimedia Object ID</th>
<th>User Query</th>
<th>Query Result</th>
<th>Rank</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>concerto</td>
<td>found</td>
<td>1</td>
<td>0.2857</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>blue, danube</td>
<td>found</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>flight, bee</td>
<td>found</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>concerto</td>
<td>found</td>
<td>2</td>
<td>0.2857</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>piano</td>
<td>found</td>
<td>1</td>
<td>0.1</td>
<td>1</td>
</tr>
<tr>
<td>38</td>
<td>handel, piano</td>
<td>found</td>
<td>1</td>
<td>0.1667</td>
<td>1</td>
</tr>
<tr>
<td>58</td>
<td>nature</td>
<td>found</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>58</td>
<td>boy</td>
<td>found</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>71</td>
<td>think</td>
<td>found</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>71</td>
<td>me</td>
<td>found</td>
<td>1</td>
<td>0.0909</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2. Query Results After Indexing
10 Conclusions

The semantic search of multimedia objects is possible by applying the semi-automated dynamic evolutionary indexing approach that we proposed. By adopting the index hierarchy, the relative importance of a semantic meaning to a multimedia object can be quantified and ranked. Through the extensive usage of the system, the index evolves continuously by user search behaviors and their relevance feedback captured.

Since the system relies heavily on the users feedback and their professional judgments. The time for the index to reach a convergence level depends on the users’ participations and knowledge. Currently, we found that users tend to be passive in giving relevant feedback. It is impossible for users to provide feedback for every search results, although users have higher chance of giving feedback in the age of web 2.0.

In the future, we will consider the improvement on the relevance feedback mechanism and the way of automatic index enrichment. Also, we will discover the relationship of user search behaviors and the index hierarchy.

References