Indexing an Intelligent Video Database using Evolutionary Control

Ronnie Cheung

Department of Computing
Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
csronnie@comp.polyu.edu.hk

Abstract: In this paper we present the implementation of an intelligent video database using evolutionary control. By using automatic video indexing techniques, the retrieval of video segments can be performed using free natural language queries. Retrieval of video segments from a database for editing and viewing is becoming an important topic in video processing. A cinematic movie consists of video segments, which are semantically related. Current approach to video retrieval emphasize on the low level semantics such as colour and textures of the video segments. However, it is difficult for the users to formulate queries in terms of these low level features. Associated with each video segment in a movie there are video scripts. Each video script contains descriptions about the content of the video and the subtitles for the video segment. Using a database of video segments with associated textual information, it is possible to provide information for video retrieval using free natural language texts. Fully automated indexing and query processing is a key problem in text-based video retrieval. To solve the associated problems, we have implemented an Automatic Video Indexing System (AVIS) using information retrieval and machine learning techniques. The system was tested using the original movie scripts from “STAR WARS – return of the JEDI” (139 movie segments) and “Star Wars – A New hope” (476 movie segments). We have formally evaluated the system using formal precision and recall measures with a fully automated indexing system. The system is able to achieve good precision-recall values. The results show that information retrieval and machine learning techniques can be applied to video information systems effectively.

Keywords: Genetic Algorithms, Video Indexing, Information Retrieval

Received 15 Jan. 2003; Revised and accepted 23 Feb. 2003

1. Introduction and Motivation

Video indexing and retrieval are important topics in video data management. The major characteristic of video data comes from the video segments and the associated semantics. This makes video data quite different from other type of data, such as text images. Content-based video indexing and retrieval techniques (Amato, Mainetto, & Savino, 1998; Yaginuma, & Sakauchi, 1996) that use low level video features such as color and textures are not sufficient to represent the high level semantics of video segments. This semantic mismatch between human understanding of video programs and the low-level video features is the major issues in indexing and retrieving video data in a high level manner. When trying to retrieve a particular video segments, the user may not be able to describe clearly the low level video features exactly. Instead, the user is able to describe the semantic contents of the video using English words. For example, to retrieve the video segments about robots, the user may construct the query “retrieve all segments about robots”. The major problem is the time consuming process of identifying and retrieving the appropriate video segments with respect to some user requirement. In the absence of an index structure, browsing and searching for relevant information in a long video file become difficult.

Movie segments are annotated with descriptions. When the movies are being produced, scripts are also available for describing the semantic details of the movie segments. Indexing and retrieval using free text annotation is an alternative approach, and is complementary, to content-based indexing using time/frame, color and graphic information. The major objective of our work is to develop a system for retrieving video segments based on semantic descriptions and annotations of video segments. Our approach is also based on text descriptions of the video segments, but an automatic indexing system is applied to generate an index for the intelligent video database.

Movies are classified into one of the several movie types such as love story, adventure, action and science fiction. We have preformed a preliminary analysis of automatic indexing using different movie types. From our preliminary experiments on automatic indexing, keywords-based retrieval techniques do not show significant results for indexing and retrieval for certain movie types, such as love story. Therefore, our focus is specific movie types such as science fiction. We have shown that machine learning techniques can be applied to such kind of movie effectively with different movies like “STAR WARS – return of the JEDI” (139 segments), and “Star Wars – A New Hope” (476 segments). We assume that each movie is pre-segmented according to the characteristics of individual video segments. Existing segmentation techniques cannot accurately determine the
segment boundary, especially when the camera is moving between different characters in a scene. Therefore, video segmentations are performed manually based on the original scripts of the movie.

2. Overview of Automatic Indexing

Automatic indexing is a well-studied subject in information retrieval (IR). Automatic indexing is the process of identifying text items to generate lists of index terms. The fundamental processes in automatic indexing compose of index terms identification and generation. Index terms identification from text documents is the process of features extraction or features selection of the documents. It is the process of selecting a subset of \( d \) features from an original set of \( D \) features, where \( d < D \). Instead of employing manual approach of features extraction, automatic feature selection is employed. In order to automate this process, a number of text processing techniques are employed:

(i) Lexical Analysis: The first stage of automatic indexing is lexical analysis. Lexical analysis is the tokenization process that converts long streams of characters in a text document into a stream of words or tokens. Tokens are collections of characters with collective significance. At the end of lexical analysis, a document is transformed into a stream of words.

(ii) Stop Words Removal: Stop words removal is the process of removing the set of non-content-bearing words (such as “a”, “an”, “the”, etc) from the stream of words produced by word extraction (Van Rijsbergen, 1979). The non-content-bearing words are called stoplist or stop words. In automatic indexing, candidate index terms found in stoplist are removed for subsequent stages since these non-content-bearing would deteriorate the quality of the index terms.

(iii) Stemming: Stemming, or conflation, is the process of combining different morphological variants of the same word into a single canonical form called stemming. A word can have a number of morphological variants. To take an example, the word “build” may exist in its other morphological variants such as “building”, “builds”. These morphological variants are different word form but represent the same idea. In automatic indexing, these morphological variants are undesirable and should be conflated to its canonical form.

(iv) Statistical Analysis: Statistical analysis is the process of analyzing the frequency of word occurrence as a means of a measurement of word significance in classical information retrieval systems. With the lower and upper cut-off, the words above the upper cut-off were considered to be common and those below the lower cut-off were considered to be rare. Words outside the lower and upper cut-off were treated as insignificant and would be ignored or even removed. Let \( \mu \) and \( \sigma \) be the mean and standard deviation of the occurrence frequencies respectively. Let \( \alpha \) and \( \beta \) be the lower and upper cut coefficient. The words having occurrence frequencies within the interval \( \mu - \alpha \sigma \) and \( \mu - \beta \sigma \) will be considered as significant words and be used in the subsequent stages in automatic indexing as the profiles of document.

In video indexing applications such as the indexing of semantic movies, a hierarchical layer of indexes is required because a movie script has a large number of segments. Searching can be guided by searching through clusters of video segments. Cluster analysis is a statistical technique employed to generate a category structure. Items are assigned to automatically created groups based on a calculation of the degree of association between items and groups. The items have a high degree of association form groups of clusters whereas items in different groups should have low degree of association.

3. Systems Overview

In AVIS the complete video is segmented manually by referring to the original movie script. The major components of the system are video indexing, and video retrieval. The original movie script segments are indexed using the automatic text indexing module. The systems architecture of AVIS is shown in figure 1.
3.1 The Index Structure

The index structure employed in this system is a hierarchical layer of content index. The basic index structure composed of both base segments and index segments where base segments represent leafs of the index tree and index segments represent nodes of the index tree respectively. Base segments are the bottom layer of content index of the index hierarchy. A base segment composed of three parts—video data, index terms, and script, as shown in figure 2. Firstly, in a base segment, there is video data - the video segment that contains a scene in the original video program. Secondly, a script is the segment of the original script of the video program describing the video segment. Finally index terms are the profile of the script segment generated in the automatic text indexing (ATI) module.
An index segment composed of three parts—super-segments, index terms, and sub-segments as shown in figure 3. Firstly, super-segments are the reference of the index segments that is one level higher than that of the current base/index segment. Secondly, sub-segments are the reference of the base/index segments that is one level lower than that of the current index segment. Finally, the index terms are the profile of the sub-segments and are generated using a genetic algorithm (Goldberg, 1989) that selects those best index terms among the index terms of the sub-segments.

![Fig 3. An Index Segment](image)

The content index is composed of two types of segments: base segments, and index segments. In the process of index generation, base segments are compared. Those base segments exceeding certain similarity will be grouped together to form index segments on top of the base segments. Index segments with certain similarity will be grouped together again to form another level of content index on top of the current content index. This process generates the content index in a bottom-up manner and, finally a hierarchical layer of content index is built.

A base segment composed of video data, script, and index term. Users input both the video data. The script and the index terms are automatically generated in the Automatic Text Indexing (ATI) module. The Base Segments Generation (BSG) process simply encapsulate the video data, script, and index terms that is generated in the ATI module by using the annotated video script.

The Automatic Text Indexing (ATI) module identifies the index terms of input script. The script entering the ATI will pass through four stages in ATI. Firstly, the script is tokenized by the lexical analyzer. The lexical analysis simply recognizes tokens/words in the character stream from the input script. Secondly, the tokens/words extracted by the lexical analyzer will be checked against a stoplist that contains stop words. Stop words are words carry little meaning. The tokens retained in the stop words removal process are considered as candidate index terms of the input script. Thirdly, candidate index terms can have different morphological variants and they are inflated to a single canonical form in the stemming process.

Finally statistical analysis techniques are adopted to identify index terms that will be combined with video data and script to form base segments. In this process, a histogram representing the occurrence frequencies of each stemmed index term is established. Each term will be considered as a potential index term if its occurrence frequencies is with in the interval \( \mu - \alpha \sigma \) and \( \mu - \beta \sigma \), where \( \mu \) and \( \sigma \) be the mean and standard deviation of the occurrence frequencies respectively, and \( \alpha \) and \( \beta \) be the lower and upper cut coefficient. After all, a set of index terms \( A = \{a_1,a_2,\ldots,a_n\} \) is generated. For each base segment, \( B_i \), a set of index terms \( K_i \subseteq A \) will be generated in ATI module. Each index term generated from the script of base segment \( B_i \) will belong to \( K_i \) if it belongs to \( A \). In this manner, the index terms for each base segment constitute a layer of content index to a base segment.
3.2 Hierarchical Layer of Content Index Generation

The generation of the hierarchical layer of content index is a bottom-up grouping process of base/index segments. Starting with the base segments, each pair of base segments $B_i$ and $B_j$, where $B_i$ is the $i^{th}$ base segment, the Jaccard’s coefficient $f(K_i, K_j)$ is calculated using the following formula (Van Rijsbergen, 1979):

$$f(K_i, K_j) = \frac{|K_i \cap K_j|}{|K_i \cup K_j|}$$

The following algorithm from (Lau, Leong, & Si, 1995) is used for generating a hierarchical layer of content index to the original video segments:-

All base segments with association value greater than a threshold, $\varepsilon$, are grouped into a collection. A new super-segment is formed for each collection, modeling a super-sub segment relationship. A base segment can have multiple super-segments if it belongs to multiple groups. For each newly created super-segment $S_i$, a genetic algorithm (Chen, 1994; Maarek, Berry, & Kaiser, 1991) is adopted to identify a set of best index terms from the index terms, $K_i$, $K_j$, ..., $K_n$, of its sub-segments, $B_i$, $B_j$, ..., $B_n$. A second level of content index is generated and the newly created super-segments are called index segment, $S_{i,j}$, where $l$ is the level of the content index and $i$ is the $i^{th}$ index segment. With the index segment in the second level, the association values $f(K_{i,l}, K_{i,j})$ between any two index segments $S_{i,j}, S_{i,j}$ in the second level can be calculated. Those index segments in the second level with their association values exceeding the threshold, $\varepsilon$, are grouped into a collection. A new super-segment, index segment, $S_{i,j}$, is formed for each collection. Another higher level content index can be generated recursively, until all the association values between any two segments within a particular level fall below $\varepsilon$, or only one segment remains in that level. Following this approach, a hierarchical layer of content index to the original video segments is created (as shown in figure 4).

![Figure 4. A hierarchical layer of content index](image-url)
4. Index Terms Generation

Let \( S_{l,i} \) be the \( i \)th segment at level \( l \), where \( l=0 \) if \( S_{l,i} \) is a base segment and \( l>0 \) if \( S_{l,i} \) is an index segment, and \( K_{l,i} \) be the index terms of \( S_{l,i} \). The initial set of segments (base segments) of the video can be denoted by \( S_{1,1}, S_{1,2}, \ldots, S_{1,n} \) where \( n \) is the total number of segments. Adopting the genetic algorithm (Chen, 1994; Maarek, Berry, & Kaiser, 1991) to generate index terms for a segment, \( S_{l,i} \), which contains \( m \) sub-segments \( S_{l-1,1}, S_{l-1,2}, \ldots, S_{l-1,m} \), with keywords \( K_{l-1,1}, K_{l-1,2}, \ldots, K_{l-1,m} \), five steps are required—reference set generation, reproduction, crossover, mutation, and iteration.

1. Reference Set Generation

Reference is a set of reference index terms. The reference set \( R_{l,k} \) of \( S_{l,k} \) is formed by the union of all the index terms of its sub-segments, \( K_{l-1,1} \cup K_{l-1,2} \cup \ldots \cup K_{l-1,m} \). Each sub-segment \( S_{l-1,i} \) will be associated with a binary vector of length \( |R_{l,k}| \) with respect to \( R_{l,k} \), indicating the presence or absence of each distinct keyword in \( K_{l-1,i} \). This provides an encoding scheme for the chromosomes. For example, if \( K_{1,1}=\{\text{Falcom, fighters, explosions}\} \), \( K_{1,2}=\{\text{explosions}\} \), and \( K_{1,3}=\{\text{fighters, dungeon, rescue}\} \), then \( R_{1,1}=\{\text{Falcom, fighters, explosions, dungeon, rescue}\} \). The binary vectors (chromosomes) corresponding to \( R_{1,1}, R_{1,2}, R_{1,3} \) are 11100, 00100, and 01011 respectively.

2. Reproduction

A new set of \( m \) sub-segments is selected from the original set of sub-segments of \( S_{l,k} \), contributing to the generation of index terms for \( S_{l,k} \). Roulette wheel selection method is adopted for selection: for each sub-segment, \( S_{l-1,i} \), will have a probability \( p_{il} \) of being selected is proportional to its fitness value, \( f(K_{l-1,i}, R_{l,k}) \), with respect to the reference set \( R_{l,k} \). In this selection scheme, a sub-segment with high fitness value might have a multiple number of times to be selected while a sub-segment with low fitness value might have few chances to be selected or even have not been selected. In the previous example, \( f(K_{1,2}, R_{2,1}) = 1/5 \) and \( f(K_{1,3}, R_{2,1}) = 3/5 \). Since \( S_{1,3} \) has a larger fitness value than \( S_{1,2} \), \( S_{1,3} \) has a larger probability of being selected than \( S_{1,2} \). At this stage, a new set of sub-segments \( \{S'_{l-1,1}, S'_{l-1,2}, S'_{l-1,3}, \ldots, S'_{l-1,m}\} \) and their index terms \( \{K'_{l-1,1}, K'_{l-1,2}, K'_{l-1,3}, \ldots, K'_{l-1,m}\} \) are generated.

3. Crossover

A random number \( r \) between 0 and 1 is generated for each sub-segment \( S'_{l-1,i} \). If \( r < p_{c} \), where \( p_{c} \) is the probability of crossover, \( S'_{l-1,i} \) will be selected for performing crossover operation. The binary vector of the two sub-segments selected for crossover will be mated randomly. At this stage, another new set of sub-segments \( \{S''_{l-1,1}, S''_{l-1,2}, S''_{l-1,3}, \ldots, S''_{l-1,m}\} \) and their index terms \( \{K''_{l-1,1}, K''_{l-1,2}, K''_{l-1,3}, \ldots, K''_{l-1,m}\} \) are generated.

4. Mutation

Another random number \( r \) between 0 and 1 is generated for every entry of the binary vector for each sub-segment \( S''_{l-1,i} \). If \( r < p_{m} \), where \( p_{m} \) is the probability of mutation, the entry is flip-flopped during the process of mutation. At this stage, another new set of sub-segments \( \{S'''_{l-1,1}, S'''_{l-1,2}, S'''_{l-1,3}, \ldots, S'''_{l-1,m}\} \) and their index terms \( \{K'''_{l-1,1}, K'''_{l-1,2}, K'''_{l-1,3}, \ldots, K'''_{l-1,m}\} \) are generated.
(5) Average Fitness Calculation and Iteration

The average fitness value between the index terms of each new sub-segment and the index terms in $R_{i,k}$ is calculated. Step (2) to (4) are repeated until the average fitness values across several subsequent iterations converge. The union of the keywords for most recently generated sub-segments will be assigned to segment $S_{i,k}$.

![Diagram of the genetic algorithm](image)

Figure 5. Index terms generation.

Figure 5 shows the detailed procedures for implementing the genetic algorithm for generating index terms that corresponds to a super-segment. In the above procedure, the fitness of the population is obtained by calculating by the average correlation (using Jaccard’s coefficient) between the terms in the super segment and the corresponding segments. Initially the average fitness of the population starts to grow gradually. After certain generations, the growth of the average fitness of the population reduced. Subsequently, the value of the average fitness converges and remains relatively constant and the genetic algorithm stops. Figure 6 shows that the fitness of the populations corresponding to a super-segment indexing converges at an average fitness about 0.7 and the populations are dominated by chromosomes with the same genotype. Note that in the indexing process, a hierarchical layer of content index is built in a bottom-up manner. When creating new super-segments, those index terms of the sub-segments selected by the genetic algorithm are considered to represent index terms that can best correspond to the newly created super-segments. The index terms generated for each super-segment are evaluated to be of high quality. These keywords that are used for navigation down to the related lower levels index structure during the searching and browsing process.

We also tested the effect of various choices for similarity threshold on the number of levels of content index generated. To test the effects of similarity threshold on the number of levels of index developed by the system, we build the index by using different similarity threshold values for $\epsilon$. Figure 7 shows the effects of assigning different values for the similarity threshold, $\epsilon$, on number of levels of the content index generated by the system. The results show that increases in similarity threshold reduce the number of levels of the content index generated. For the threshold value 0.15, there are 22 levels of layers generated. With a threshold value of 0.2, 8
levels of layers are built. For the threshold value corresponding to 0.25 and 0.3, there are 2 and 3 levels of layers generated respectively. The experiment results reveal that $\varepsilon = 0.2$ is a typical choice for the similarity threshold.

![Average Fitness vs. Generation](image)

**Figure 6.** The average fitness vs. generation

Various choices for the similarity threshold, $\varepsilon$, affects the number of level of the content index. If the threshold is over estimated, few or even no segments have enough association/similarity to produce super-segments/collections. With index generated from an over estimated threshold, few segments would be grouped and most of the association between segments, regardless of their usefulness, are ignored and treated as different unrelated segments. This produces an index with few layers of content index and few association linkages between segments. This kind of index produces little directions to guide searching and browsing. Hence, with an over estimated similarity threshold, the lost of useful information would increase.

If the threshold is under estimated, segments with low degree of association are grouped together to produce super-segments/collections. When the index is built with an under estimated threshold, distinctiveness of different segments are lowered. This produces an index with many layers of content index and huge linkages between segments. These kinds of index provide much useless directions for the searching and browsing process. Hence, with an under estimated threshold, the quality of the index generated will be deteriorated by useless and misleading linkages between segments. With a properly tuned threshold, the index generated provides a good content index.

![Similarity Threshold vs. No. of Levels of Index](image)

**Figure 7.** Similarity threshold vs. number of levels of content index
5. Systems Implementation and Evaluation

The major components of the system are the Description Parser, Description Index Generator, an Index Explorer and a Search Engine. The description parser identifies and generates candidate index terms of the script of the video, including tokenization and conflation of the script. The main function of the description index generator is to build the hierarchical layer of content index of the selected video program. The Index Explorer enables users to browse and navigate video segments by adopting the hierarchical layer of content index. Users can browse through the content index and the index terms and script of the selected segment will be displayed. For those base segments, users can view the video segment as well. A sample screen layout of the description index explorer with a hierarchical layer of content index is shown in figure 8.

![Sample screen layout of the description index explorer](image)

Figure 8. A sample screen layout of the description index explorer

The search engine allows the users to retrieve video segments by using free natural language text. Users should firstly select the video program they would like to search. Search parameters are entered by the users and these parameters are used to direct the behavior of the search engine. Search parameters consist of four parts—search text, similarity threshold, number of items to be retrieved, and the area to be searched. Results of the search are ranked and displayed on the user interface. Users can dynamically select a field of the result to rank. In figure 9, the user selects “x-wing and skywalker”. A list of related video segments are listed in the order of similarity. The user can click on the “Play Video” button to play the selected video.
For a given query, we evaluated whether a video segment is relevant. For the evaluation, we define a video segment is relevant if the user can identify a scene in the retrieved video segment relevant to the query. By this comparison, the retrieval effectiveness can be computed according to two commonly used measures, namely precision and recall. Precision measures the accuracy of the retrieval result as indicated by the proportion of retrieved segments that are relevant. Recall measures how extensive the retrieval result is, as indicated by the proportion of relevant documents retrieved. They are defined as follows:

\[
\text{precision} = \frac{\text{Number of relevant retrieved segments}}{\text{Total number of segments retrieved}}
\]

\[
\text{recall} = \frac{\text{Number of relevant retrieved segments}}{\text{Number of segments in the database that are relevant to the query}}
\]

For the set of retrieval results corresponding to tested queries, a set of precision and recall values is computed. Based on the set of pairs of precision and recall values, a precision-recall graph is shown in figure 10. The system performs well in terms of precision and recall measures. Compared with other content-based retrieval systems, our system also provides a user-friendly environment for querying using semantic features of the video segments.
6. Conclusions

The effectiveness of applying genetics algorithms to automatic video indexing for the intelligent database was tested experimentally using AVIS. The videos “STAR WAR – return of the JEDI” (139 movie segments) and “Star Wars – A New hope” (476 movie segments), and the associated script was used for the evaluation. The video is segmented to non-overlapping segments according to its script manually. After the lexical analysis and stop words removal process, all keywords are stemmed, and a statistical analysis is performed for selecting candidate index terms for the system. The system was tested using the original movie scripts from the videos. A fully automatic index system is generated by AVIS for the movie segments. Terms corresponding to 11589 annotated free text descriptions of a video are generated for the super segments for indexing. The super-segments, and the corresponding index terms are generated by a genetic algorithm in a bottom-up manner. The index terms generated for each super-segment are evaluated to be of high quality. These keywords are used for navigation through the lower levels index structure during the searching and browsing process. We have formally evaluated the system using formal precision and recall measures using a fully automated indexing system. The system is able to achieve good precision-recall values. The results show that information retrieval and machine learning techniques can be applied to video information systems effectively.

7. References


