Hindawi Publishing Corporation
Advances in Multimedia
Volume 2011, Article ID 310762, 18 pages
doi:10.1155/2011/310762

Research Article

Utilizing Implicit User Feedback to Improve Interactive Video Retrieval

Stefanos Vrochidis,[1,2] Ioannis Kompatsiaris,[1] and Ioannis Patras[2]

[1] *Centre for Research and Technology Hellas, Informatics and Telematics Institute, 6th Klm Charilaou-Thermi Road, 57001 Thessaloniki, Greece*
[2] *Queen Mary, University of London, Mile End Road, London E1 4NS, UK*

Correspondence should be addressed to Stefanos Vrochidis, stefanos@iti.gr

Received 1 September 2010; Accepted 3 January 2011

Academic Editor: Andrea Prati

This paper describes an approach to exploit the implicit user feedback gathered during interactive video retrieval tasks. We propose a framework, where the video is first indexed according to temporal, textual, and visual features and then implicit user feedback analysis is realized using a graph-based methodology. The generated graph encodes the semantic relations between video segments based on past user interaction and is subsequently used to generate recommendations. Moreover, we combine the visual features and implicit feedback information by training a support vector machine classifier with examples generated from the aforementioned graph in order to optimize the query by visual example search. The proposed framework is evaluated by conducting real-user experiments. The results demonstrate that significant improvement in terms of precision and recall is reported after the exploitation of implicit user feedback, while an improved ranking is presented in most of the evaluated queries by visual example.

1. Introduction

In the recent years, the rapid development of digital technologies has led to the growing storage and processing capabilities of computers, as well as to the establishment of fast and advanced communication networks. Taking also into account the low cost of image and video capturing devices and the deep penetration of Internet in today's communities, large quantities of audiovisual content has become available and accessible worldwide. The availability of such content and the increasing user need of searching into multimedia collections place the demand for the development of advanced multimedia search engines; therefore, video retrieval remains one of the most challenging tasks of research. Despite the recent significant advances in this area, further advancements in several fields of video retrieval are required to improve the performance of current video search engines. More specifically, major research breakthroughs are still needed in the areas of semantic and interactive search possibly using multimodal analysis and retrieval algorithms, as well as relevance feedback [1].

The state-of-the-art video retrieval systems incorporate and combine several advanced techniques including text retrieval and visual content-based search, in order to support the user in locating video clips that meet their demands. One of the main challenges faced by these approaches is to generate efficient representations and descriptions of the video source. The initial step towards this direction is the video segmentation and indexing into smaller video shots [1]. Based on the audiovisual information, it is possible to extract low level features [2] for each shot; however, due to the well known problem of the semantic gap, it is difficult to associate them with human understandable concepts. Combination and fusion of heterogeneous information (e.g., visual, textual, and motion) have been considered as the first step towards the solution of this problem and promising results have been presented both in image and video retrieval domains [3–6]. However, the progress in the representation

and indexing of the multimedia content has not yet managed to overcome the semantic gap.

An alternative way to bridge the semantic gap is to utilize the implicit and explicit feedback provided by the users [7] of a video search engine. During an interactive video-retrieval task, multiple search sessions take place, in which the user submits queries, browses the video information or even provides explicit feedback on presented results. Relevance feedback (RF) mechanisms in information retrieval were devised as complementary methods to further improve the performance of a system by requesting explicit feedback from the user (i.e., to identify positive and negative examples) that would guide machine learning techniques. Despite the promising results in image and video retrieval [8–10], RF-based functionalities are not very user popular, as their main drawback is that users are usually reluctant to provide explicit information [11, 12]. This is also reflected in popular web search engines (e.g., Google, YouTube, etc.), which provide user interfaces that do not request or require explicit user feedback during search sessions. Motivated by this fact, we propose to exploit the implicit user feedback during the search process, in order to gain new knowledge about the content and establish semantic relations between the involved multimedia items. This information can be utilized both for generating recommendations, as well as for improving the performance of existing content-based retrieval modalities.

In this work, we consider as implicit user feedback any action or navigation behavior of the user during interactive video retrieval tasks, including mouse movements and clicks as well as keyboard inputs and keystrokes. The main advantage of using implicit techniques is that they do not require any explicit feedback from the user side. Although implicit information is in general considered to be less accurate than explicit [13], large quantities of implicit data (e.g., log files in web search engines) can be gathered at no extra effort to the user. Recent approaches in interactive video search attempt to exploit past user interaction either for performing retrieval [7] or for complementing existing content based search modalities [14]. In this context, we propose a video retrieval framework, which supports video analysis, as well as implicit user feedback recording and processing. Then, we provide recommendations based on past user interaction and we offer a hybrid visual search modality by combining heterogeneously extracted information (i.e., implicit feedback and visual features) by employing machine learning methods.

Video processing is realized with the application of text, audio, image, and video analysis, including video to shots segmentation, keyframe extraction and visual features generation, as well as Automatic Speech Recognition (ASR) and conversion to text. On the other hand, we exploit the implicit user feedback, in order to initiate semantic relations between the video segments. This is performed by introducing implicit interest indicators for video search and then by constructing a semantic affinity graph, which is utilized to generate recommendations in the following two steps. First, an action graph that describes the user navigation pattern is generated by employing a novel methodology that defines search subsessions (i.e., parts of sessions in which the user searches a specific topic) based on query categorization. Then, a set of action graphs is converted to a single weighted graph by aggregating the action graphs and assigning weights to the user actions that quantify the implicit interest indicators. In order to provide recommendations, we employ a distance-based algorithm to rank the graph nodes. In the case of a query by visual example, this graph is utilized in a similar way in order to define positive and negative examples. The latter are merged with a set of visually similar and dissimilar examples based on visual features, in order to construct a training set, which is used to train a support vector machine (SVM) classifier that reranks the results of the visual search.

This framework is realized in an interactive video search engine, which supports basic retrieval functionalities including text, visual, and temporal search. The search engine is used for the evaluation of the approach by conducting real-user experiments in 3 phases: first, a baseline system that supports only video analysis retrieval options is used by the users, and their actions are being recorded; then, different users are searching for topics that are slightly different than the baseline ones using the enhanced version of the search engine, which exploits also user implicit feedback, in order to evaluate the recommendations; finally, in the third phase, different users are recruited to evaluate the reranking of the visual results.

This paper discusses in more detail and integrates the works presented in [15, 16] into a solid framework, while an extended evaluation through additional experiments and comparisons between the techniques introduced in the aforementioned articles are presented. The research novel contributions of this work are summarized in the proposed methodology of graph analysis based on query categorization and the definition of subsessions, as well as the methodology for combining visual features with implicit user feedback. To the best of the authors knowledge, this is one of the first attempts to combine patterns of past user interaction with visual features.

This paper is structured as follows: Section 2 presents the related work, while in Section 3, we introduce the SVMs employed in the reranking of the query by visual example results. The video indexing and retrieval framework is presented in Section 4, while Section 5 deals with the video content analysis. Section 6 describes the processing of user implicit actions based on a graph approach, and Section 7 presents the methodology for combining visual features with graph-structured implicit user feedback. Section 8 demonstrates the implemented search engine and the user interaction modes, while the experimental results and the evaluation are presented in Section 9. Finally, Section 10 concludes the paper.

2. Related Work

In general case retrieval tasks, the implicit user feedback can be divided into two main categories: the query actions and the physical user reactions. The first category includes the patterns of user interaction with the search engine, as series

of mouse movements and clicks, shot selections, key strokes, and keyboard inputs, while the second includes physical user unconscious behavior as eye movements (e.g., [17]), heart rate, and brain neuron reactions that can be gathered with electroencephalography (e.g., [18]). On the one hand, the feedback of the first category can be easily gathered even during a web search session, while physical reactions can be recorded with the aid of special wearable devices or other sensors (e.g., cameras) capturing and analyzing user behavior. In this work, we will focus on exploiting the user feedback that falls into the first category.

Implicit feedback approaches based on the user interaction with search engines have been effective in the context of textual retrieval, where they were mostly employed for query expansion and user profiling, in order to retrieve, filter, and recommend items of interest [19]. The "implicit interest indicators" were introduced in [20] as a definition of specific user actions that can be considered as meaningful implicit feedback. In [21], the authors performed a comparison between an explicit and an implicit feedback system concluding that there were not significant differences between these two systems and that substituting the former with the latter could be feasible.

In another interesting work [11], the author attempts to automatically optimize the retrieval quality of search engines using clickthrough data. The proposed method utilizes clickthrough data (by exploiting the query-log of the search engine) for training. Taking a support vector machine (SVM) approach, this paper presents a method for learning retrieval functions. More specifically, the clickthrough data are translated into ranking user preferences, and then they are used to train a retrieval function. The implemented SVM in this case has been especially designed in order to be trained by such rankings, which reflect related user preferences (i.e., one option is better than another).

In [12], the authors propose to detect "query chains" (i.e., a sequence of queries) and then learn a retrieval function using SVMs. The authors demonstrate a simple method for automatically detecting query chains in query and clickthrough logs. These data are used to infer preference judgments regarding the relative relevance of documents both within individual query results and between documents returned by different queries within the same query chain. The method used to generate the preference judgments is validated using a controlled user study. A ranking SVM is adapted to learn a ranked retrieval function from the preference judgments. The results demonstrate significant improvements in the ranking given by a normal search engine.

Implicit feedback techniques have not been fully explored in the multimedia domain [22]. In text retrieval, the usual implicit information that can be taken into account is the user selection (i.e., the user clicks on an interesting link or textual description to view the complete document), while in video retrieval, we have multiple interactions between the user and the system, which could be utilized to provide meaningful feedback. The main idea to exploit the user feedback during video retrieval interactive sessions is to extend the idea of "query chains" [12] and construct a graph that describes a series of user actions. Such a graph is transformed to a weighted graph by aggregating the links between the same nodes, and weights are introduced based on the different actions taken into account. Recent works [7, 23] employ the aforementioned technique to deal with user clicks.

In [7], the authors propose to use community-based feedback mined from the interactions of previous users of a video retrieval system, which is based on Okapi BM25 retrieval model supporting text queries to aid users in their search tasks. This feedback is the basis for providing recommendations to new users of the video retrieval system. This is performed by representing all user interactions with a weighted graph. Then, this implicit information is aggregated from multiple sessions and users into a single representation, thus facilitating the analysis and exploitation of past implicit information. In [23], the authors evaluate 4 different algorithms that can be applied on such weighted graphs to provide recommendations. However, these works consider only textual queries, while basic video retrieval options as visual and temporal based search are ignored. In addition, fusion or combination of implicit feedback data with the content-based approaches is not attempted.

In another work [14], a video retrieval system is presented, which employs relevance feedback and multimodal fusion of different sources (textual, visual, and clickthrough data), in order to generate recommendations for the user. In this approach, the textual, visual, and aural data of the video shots are processed separately and compared with the selected video document. Then, these results are fused. A further adjustment of the fusion weights is performed with the aid of the click through data, which denote the interest of the user to a specific document based on the time he/she watched the video shot. The approach of fusing content analysis information with the implicit feedback seems to be very interesting and promising; however, in this specific work, the implicit information is not very deeply exploited, as the sequence of query actions is not taken into account, failing in that way to semantically interconnect subsequent queries and shots. In comparison to the proposed approach, this work employs implicit user feedback for adjusting weights, in order to fuse results from different search modalities (e.g., text and visual) and not for providing recommendations or for improving the results of a specific retrieval module (i.e., in this case visual search).

3. Support Vector Machines

Support vector machines constitute a set of supervised learning methods, which analyze data and recognize patterns and are employed to solve classification and regression problems. When a set of training positive and negative examples is available, an SVM training algorithm builds a model that predicts in which category a new example falls into. To achieve that, a support vector machine constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space. In general, it is assumed that the best separation is achieved by the hyperplane that has the largest distance from the nearest training datapoints of any class.

In this work, SVMs are used in the reranking of the initial results of the query by visual example, where visual and implicit feedback information are combined. We employ an SVM implementation that realizes the alternative structural formulation of the SVM optimization problem for conventional binary classification with error rate described in [24]. For a given training set $(x_1, y_1), \ldots, (x_n, y_n)$ with $x_i \in R^N$ and $y_i \in \{-1, +1\}$, training this binary classification SVM solves the following optimization problem, which was proposed for predicting structured outputs and optimizing multivariate performance measures like F_1-score or the precision/recall break-event point [25]

$$\text{mean}_{w,\xi\geq 0}\frac{1}{2}w^T w + C\xi, \tag{1}$$

$$\text{subject to}: \forall c \in \{0,1\}^n : \frac{1}{n}w^T\sum_{i=1}^{n}c_i y_i x_i \geq \frac{1}{n}\sum_{i=1}^{n}c_i - \xi, \tag{2}$$

where C is the capacity constant and w a parameter vector. This approach has 2^n constraints, one for each possible vector $c = (c_1, \ldots, c_n) \in \{0,1\}^n$, and it has only one slack variable ξ that is shared across all constraints. The algorithm that is employed to solve the aforementioned classification SVM optimization problem is an adaptation of the cutting-plane algorithm. This algorithm iteratively constructs a sufficient subset W of the set of constraints. Starting with an empty set of constraints W, in each iteration, it first computes the optimum over the current working set W (i.e., $w = 0$ and $\xi = 0$ in the first iteration), and then it finds the most violated constraint in the optimization problem and adds it to the working set W [24].

The main reason for selecting this specific SVM implementation in this approach was the very fast performance it demonstrates, which was important for performing the ranking optimization at real time during the query process using an adequately large training set.

4. Video Retrieval Framework

The overall video indexing and retrieval framework proposed in this approach is illustrated in Figure 1. At the bottom part, video analysis is performed, including video segmentation to shots, textual analysis of audio information and visual processing of extracted keyframes. On the top part of the framework, the implicit user feedback is captured in log files and analyzed in order to initiate semantic relations between parts of the audiovisual content. The recommendations generation component is capable of providing suggestions by exploiting the results of implicit user feedback processing. Furthermore, the outputs of implicit feedback and video analysis are combined in a separate component, which employs training of an SVM classifier, in order to rerank the results of queries by visual example. Processing of implicit information and video content-based analysis take place off line, while the combination of visual and implicit information is realized on the fly (i.e., in real time when a new query is submitted). In the next sections, these parts will be described in detail.

5. Video Analysis

The role of video analysis part is to process the initial video source in order to generate and index efficient representations of it, which can be utilized for retrieval. Temporal-, textual-, and visual-based indexing operations are performed as described in the sequel.

5.1. Temporal Indexing. As stated in the introduction, in order to generate an efficient representation of the initial video source and index it according to temporal information, shot boundaries detection and shot segmentation steps are required to split the video into shots. In this work, shot detection is achieved by thresholding the distance between color histograms corresponding to two consecutive frames in a video [26]. Then, each video is segmented into smaller shots based on the shot boundaries detection information and the middle keyframe for each shot, which is considered as the representative one, is extracted. In that way, a temporal indexing structure is constructed, so each video can be represented by a sequence of images (i.e., one image per video shot).

5.2. Keyword Indexing. Indexing of video shots according to the associated textual information is realized following the approach of [27]. The audio information is processed off line with the application of automatic speech recognition (ASR) on the initial video source, so that specific sets of keywords can be assigned to each shot. Indexing and query functions are implemented using the KinoSearch full-text search engine library [28], which creates a binary index optimized for speed. First, stopwords are removed from keyword files followed by the application of the Porter stemming algorithm [29]. Then, term weights for each keyword are computed by applying the BM25 text algorithm [30]. Finally, the keywords are associated with the segmented shots.

5.3. Visual Similarity Indexing. The visual similarity shot indexing is realized with the extraction of low level visual descriptors following the approach of [27]. In this case, five MPEG-7 descriptors, namely, color layout, color structure, scalable color, edge histogram, homogeneous texture, are generated from the representative keyframe of each video shot and stored in a relational database. By concatenating these descriptors, a feature vector is formulated to compactly represent each keyframe in the multidimensional space. An empirical evaluation of the system's performance using different combinations of the aforementioned descriptors advocated the choice of one MPEG-7-based scheme, which relies on color and texture (i.e., color layout and edge histogram are concatenated) [27]. Distance calculation between the descriptors of two shots is performed by employing the functions proposed in the MPEG eXperimentation Model [31].

To further improve the performance of the system in terms of time response, we employ a multidimensional R-tree indexing structure, which is constructed off line using

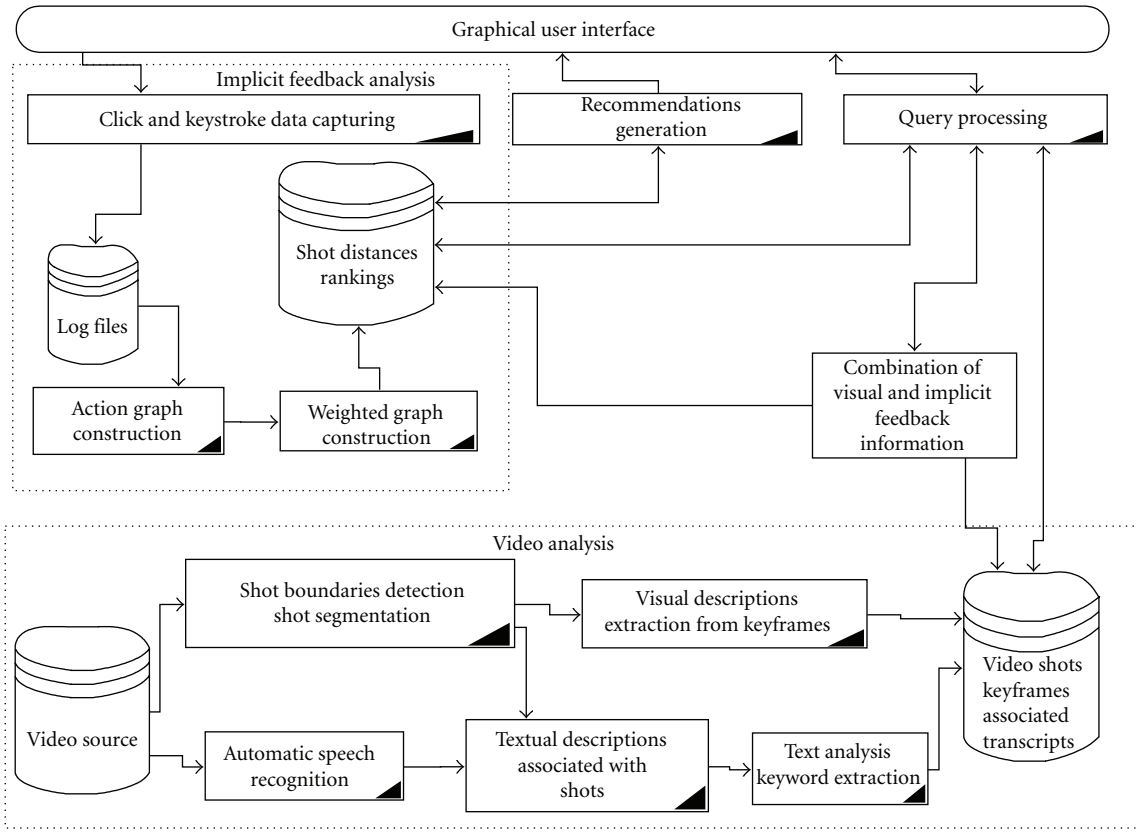

Figure 1: Video indexing and retrieval framework.

the feature vectors of all the extracted keyframes. R-tree(s) [32] are structures suitable for indexing multidimensional objects and known to facilitate fast and efficient retrieval on large scale. Principal component analysis (PCA) is also employed to reduce the dimensionality of the initial space.

In the query phase, when a query by visual example is initiated, the feature vector of the query shot (i.e., from the representative keyframe of it) Q is extracted and it is submitted to the R-tree indexing structure. The latter returns a set of not ranked K results R_V, which contains the shots $s_i, (0 \leq i \leq K)$ that are found to resemble the query one. The final visual ranking is performed by calculating the visual distances d_V between Q and all shots in R_V, so we have $d_V(Q,i) = f_V(Q,s_i)$, where f_V is the visual distance computing function and $s_i \in R_V$. The ranking for a query Q can be described as a new ordered set $RK_V = (s_a, s_b, s_c, \ldots)$, where $d_V(Q,a) \leq d_V(Q,b) \leq d_V(Q,c), \ldots$ and the cardinality of RK_V elements is K.

6. Implicit Feedback Analysis

This section presents the analysis of implicit user feedback expressed in mouse clicks and keyboard inputs. First, we introduce the implicit interest indicators that we will consider for video search, then we construct action graphs based on the user navigation patterns, and finally we aggregate them to a single weighted graph, which can be utilized to generate distances between the video shots.

6.1. Implicit Interest Indicators. Our objective is to define implicit interest indicators [20] that measure aspects of the user interaction, in order to exploit the information that the latter carries about the user's perception of the presented multimedia material.

Based on the search functionalities offered by the video analysis and indexing modules, which extend beyond the classical text-based queries and are already included in existing systems (e.g., [33]), we introduce the following minimum set of user actions that can be considered as the main implicit interest indicators for video retrieval.

(1) Text-based query (TQ): the user inserts a keyword and submits the query. In this case, we assume that this keyword satisfies the query of the user with a very high probability.

(2) Visual query (VQ): the user submits a query by visual example. We assume that when a user selects a keyframe and searches for visually similar images, then there is also interest in the example used.

(3) Side-shot query (SQ): the user selects a shot in order to view the temporally adjacent shots and the associated textual description. In that case, the user is very likely to be interested in the shot selected.

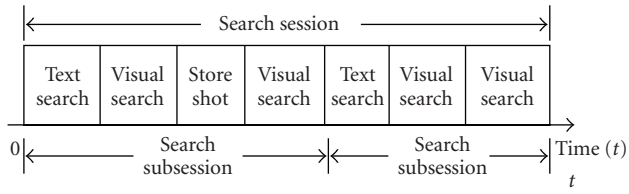

Figure 2: Search session and subsessions.

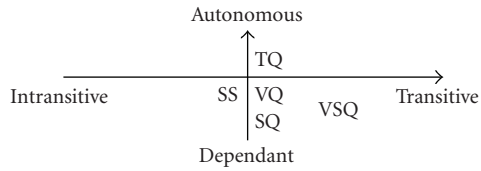

Figure 3: Classification of user actions.

(4) Video-shot query (VSQ): the user selects a shot (from the presented results of another query) and retrieves all the shots of the same video. In this case, we consider that the user is interested in the initial shot to a certain extend.

(5) Submit a shot (SS): the user marks a shot as relevant. In this case, we assume that the user is very interested in this shot. In a web search system, this action could be equivalent to watching the video shot.

6.2. Action Graph. We exploit implicit feedback information by employing an extended variation of the graph construction methodology proposed in [7]. While [7] uses a search system that supports only text-based queries, we deal with a more complex situation since the proposed interactive retrieval system supports in addition the submission of visual and temporal queries.

In order to describe better the proposed methodology, we introduce some basic definitions. We define as "search session" the time period that a certain user spends on searching. We consider as "search subsession" the time periods a certain user spends searching for a specific topic. In addition, we provide a categorization schema for the user actions during interactive video search tasks. First, we introduce the property of "transitivity", which characterizes an action based on its output. More specifically, we consider an action as "transitive", when it generates an output and so it satisfies the triplet

$$\text{input} \longrightarrow \text{action} \longrightarrow \text{output}. \tag{3}$$

On the other hand, when an action does not provide any output, (i.e., input $\rightarrow$ action) it is characterized as "intransitive". Furthermore, we classify the query actions into two main categories based on their dependency with previous actions: (a) the autonomous queries, which do not depend on previous results and (b) the dependent queries, which take as input results from previous search actions.

To construct an action graph based on the user search activity, we exploit the properties of the involved user actions. During a search session, it is possible to have a series of transitive actions, where part of the output of one action is the input for another (e.g., a result from a text search is the input for a visual search). Consequently, to create a link between two nodes of an action graph, we need to have a sequence of two actions, where at least the first one has to be transitive. During a search session, the user may search for a specific topic; however, it is possible to fire a search having a very broad or complex topic in mind or even decide to change the search topic during the session. For this reason, we propose that such sessions should not be analyzed as whole, but should be first decomposed into subsessions. Assuming that every autonomous query could initiate a different topic search, we propose a novel methodology, based on that, we divide each search session into "search subsessions" generating in that way several subgraphs and using as break points the autonomous queries.

Taking into account the corresponding functionalities of the introduced implicit interest indicators, only the text-based search can be denoted as autonomous query, while the other queries are considered as dependent. In such a case, the text-based query is utilized as a break point between the subsessions as illustrated in the example of Figure 2. It should be noted that in this analysis, the visual query (VQ) does not constitute an autonomous query, as it is based on results of previous queries. However, in the case that the insertion of a new image/video shot is supported, the VQ could be also considered as autonomous. The overall classification of these functionalities can be visualized in the two different axes of transitivity and dependency as shown in Figure 3.

In the general case, a search subsession S consists of a set of actions A_s that includes one autonomous and a number of dependent query actions. The proposed subgraph G_s is comprised by a set of nodes (i.e., shots and keywords that represent inputs and outputs of a set of actions A_s) and links that represent the corresponding actions $a_i \in A_s$, where $i \in \{1, \ldots, N_S\}$ and N_S is the cardinality of the elements of A_s. The action graph of a search session is composed of several subgraphs, which reflect the respective subsessions and have as parent nodes the autonomous queries.

These are illustrated in the example of Figure 4, where an action graph for a search session is presented. Here, the user is searching for shots, in which people sitting at a table talking are depicted. We observe that the three keywords that were used to start the search (i.e., talk, sit, and dialogue) were considered as the parents for new subgraphs. Then, we construct a single-action graph aggregating the action graphs from the different user sessions. More specifically, all the nodes from the individual action graphs are mapped to single-action graph, and then all the action edges are mapped onto the same graph, generating in that way multiple links between the nodes. We observe that the three keywords that were used to start the search (i.e., talk, sit, and dialogue) are considered as the parents for new subgraphs, which correspond to different subsessions. In this way, concepts with different semantic meaning are not interconnected (e.g., "talk" with "sit"), while keywords with similar semantic meaning (i.e., "talk" and "dialogue")

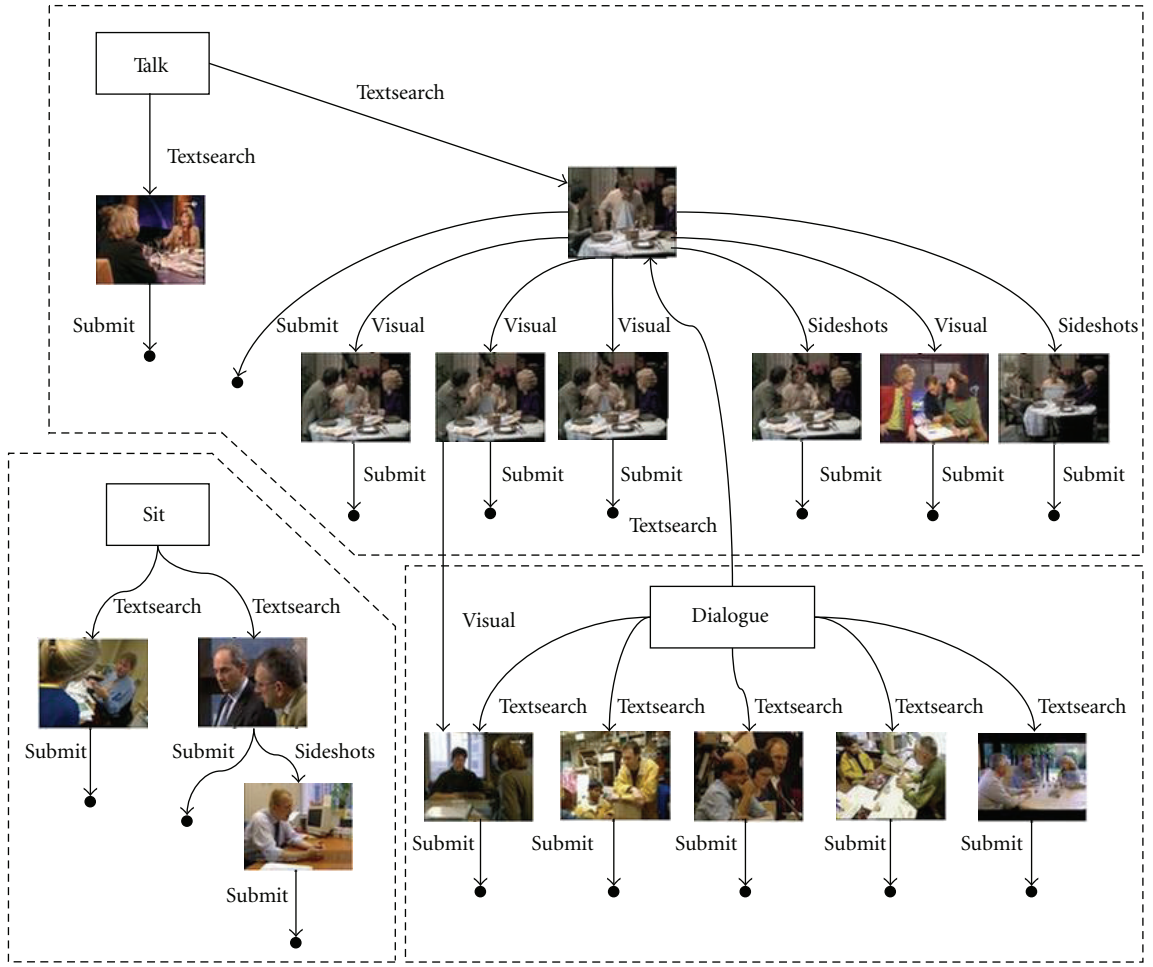

Figure 4: Action graph after user interaction.

are eventually interconnected due to the visual similarity between two shots in different subgraphs. Then, we construct a single-action graph aggregating the action graphs from the different user sessions.

6.3. Weighted Graph. After the construction of a single action graph, we generate the weighted graph in the following three steps: (a) the relevant results are linked to the parent query, transforming in that way the intransitive actions into transitive, (b) the multiple links between the same nodes are collapsed into one and (c) actions are translated into weights.

The final weight w for a link n between two nodes k and m is given by

$$w(n) = 1 - \frac{1}{x(n)}, \tag{4}$$

where $x(n)$ is the sum of weights of each action that interconnects nodes k and m. This sum is expressed as

$$x(n) = \sum_{a \in U_s} g(a), \tag{5}$$

where g is a function that maps each action to an implicit weight and a is an action that belongs to the set of actions $U_s \subseteq A_s$ that comprise the different links between the nodes k and m [7].

Table 1: Average assigned weights for each action.

Actions (a_i)	$g(a_i)$	Actions (a_i)	$g(a_i)$
Text-based query (TQ)	7.9	Visual query (VQ)	8
Side-shot query (SQ)	7.1	Submit a shot (SS)	9.1
Video-shot query (VSQ)	5.8		

Following the analysis in Section 6.1, we assign indicative values (between 0 and 10) that quantify the level of interest associated to the introduced implicit interest indicators (Table 1). In order to assign the representative weights, we asked 10 users to measure the level of interest for each search action in the range between 0 and 10. The results (i.e., the average weights for each action) of this survey are presented in Table 1. Using (4) and (5) and the defined values for g, we are able to construct the weighted graph.

Figure 5 illustrates the weighted graph that is produced after processing the action graph of Figure 4 according to the aforementioned methodology.

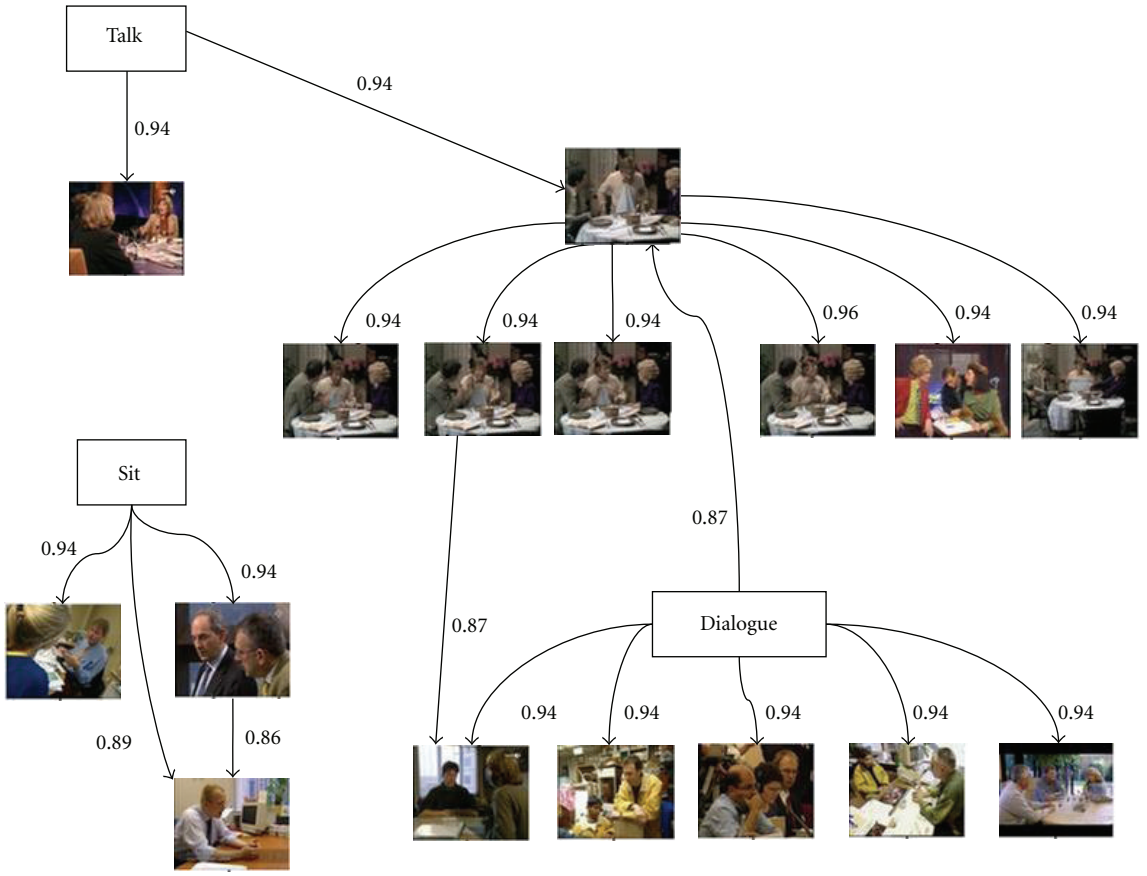

Figure 5: Weighted graph after processing the action graph of Figure 4.

6.4. Generation of Recommendations. In [23], several recommendation algorithms based on such a weighted graph were proposed. However, in most of the cases, the best performing algorithm was depending on the search topics. Here, we employ a straightforward algorithm that initiates recommendations based on the distances on the weighted graph. The latter are calculated as the shortest path between two nodes. The calculation of the distances between two different nodes in this graph was done with the application of Dijkstra algorithm [34], which computes the shorter path between two nodes. Although Floyd's algorithm [35] is usually faster for calculating all the shortest distances according to graph theory, it is better suited for more dense graphs. In our case, the produced weighted graphs are considered to be rather sparse and can be represented more efficiently with the aid of adjacency lists instead of adjacency matrices. Therefore, the proposed approach is considered scalable, since it can deal with huge and very sparse graphs, which are produced when many users and large datasets are involved.

As the calculated distance is based on implicit information, but it reveals semantic relations, we name it "implicit semantic distance". Hence, based on the shorter path approach, we can calculate the implicit semantic distance of each query Q that is represented as a node in the graph, with the rest of the nodes included in the graph. In this case, we need to notice that the query Q can be either a shot or a keyword, while the same stands for the results. Formally, we compute $d_I(Q,i) = f_I(Q,s_i)$, where f_I is the implicit distance computing function, $s_i \in R_I$, R_I is the set of M shots or/and keywords that are interconnected through links with the query Q and $1 \le i \le M$. The ranked recommendations for query Q can be described as a new ordered set $RK_I = (s_{\acute{a}}, s_{\acute{b}}, s_{\acute{c}}, \ldots)$, where $d_I(Q,\acute{a}) \le d_I(Q,\acute{b}) \le d_I(Q,\acute{c}), \ldots$ with a cardinality of M elements.

Another important functionality of the weighted graph is that it can be used to suggest new search term recommendations by calculating the distances of the input keyword term with the rest of the keywords in the weighted graph. Analogously, it can also generate related terms for a query by visual example by presenting to the user the keywords that are found to be closer in terms of distance to the query shot in the weighted graph.

7. Combining Visual and Implicit Feedback Information

The objective of this part is to rerank the initial results of the query by visual example by utilizing the weighted graph. Although the results obtained based on visual descriptors are usually quite satisfactory, in many cases, visual search fails to fetch results of the same semantic meaning confused

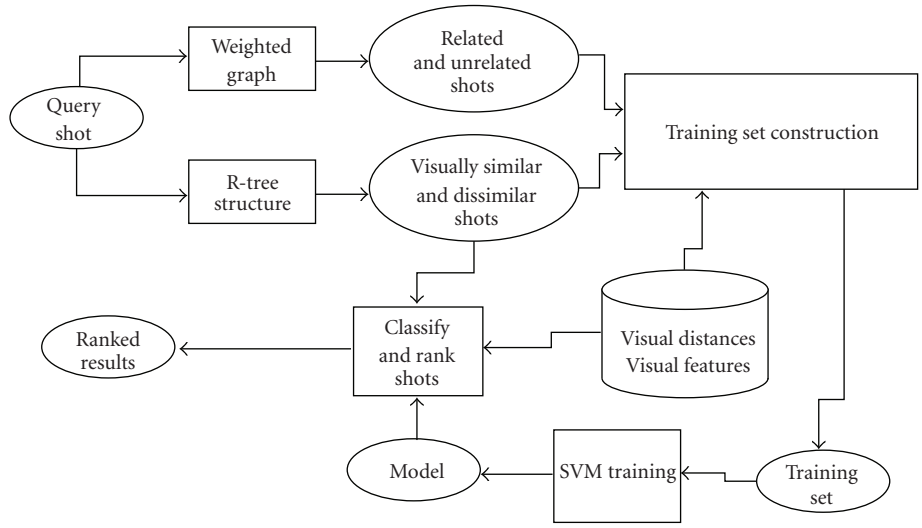

Figure 6: Algorithm to combine visual features and implicit user feedback.

by similar colors or textures of semantically irrelevant depictions. As discussed in Section 5.3, visual search is performed in the following two steps: (i) by submitting the query descriptors to the R-tree structure and (ii) by ranking the results returned calculating the distances between visual descriptors. The idea is to tune appropriately the ranking function with the aid of semantically related shots in order to emphasize more on the specific visual features that can be of importance for each query. It is expected that training a classifier with semantically positive and negative examples from the user implicit feedback could optimize the ranking function adequately. More specifically, we train a classifier for each visual example query by employing as training set a combination of visually similar and dissimilar examples, as well as positive and negative samples generated by implicit feedback information, in order to rerank the initial visual results. In Figure 6, the overall algorithm of the proposed approach is presented.

As shown in the diagram, when a query by shot example Q is submitted, we produce the following ranked datasets of results: two sets R_I and U_I (i.e., related and unrelated shots, resp.) using the weighted graph that is constructed by processing the past user interaction and one set R_V provided by the R-tree structure that includes visually related shots according to the visual features employed. Subsequently, these sets are merged in order to construct a training set T and then train a support vector machine classifier utilizing as features the visual descriptors. Finally, we employ the R_V (i.e., the set of results from visual search) as the test set, which is ranked according to the degrees of coefficients that are the output of the classifier and represent a similarity metric between each shot of the test set and the query. In Sections 7.1 and 7.2, we provide the details about the training set construction and the SVM training.

7.1. Training Set Construction. As was described in Section 3, in order to train a binary support vector machine classifier, we need to identify a proper training set $T = T_P \cup T_N$, where T_P is the set of the positive and T_N the set of the negative samples. Utilizing the weighted graph, we can extract a number of positive samples that are closer to the query shot and a number of negative samples that are placed as further as possible in this graph. Hence, we create the set of positive samples $T_{I,P} = R_{I,P}$, where for all $s_i \in R_{I,P}$, $d_I(Q,i) < d_{I,l\text{thres}}$ and $d_{I,l\text{thres}}$ is an experimentally set threshold. In the same way, we define a set of negative examples by employing another distance threshold $d_{I,h\text{thres}}$, and we consider the $T_{I,N} = U_I$, where for all $s_i \in UK_I, d_I(Q,i) > d_{I,h\text{thres}}$. In the best case, these shots should not be interconnected with the query shot in the graph (i.e. $d_I(Q,i) = \infty$). Alternatively, we can select a predefined number of negative and positive samples from the graph and apply the distance thresholds only if required.

The obvious approach could be to simply train the classifier with $T_{I,P}$ and $T_{I,N}$. However, due to the fact that implicit feedback is not always precise, such an approach would not always be efficient. On the other hand, visual search is usually capable of retrieving very similar keyframes, which demonstrate an almost zero visual distance. Therefore, in order to minimize such effects and in addition to exploit the results from visual ranking that are of good quality, we include in the positive samples the shots that are visually closer to the query example by employing an experimentally set threshold. Furthermore, we include in the negative samples some of the visual results that are very far from the query image taking into account again a visual distance threshold.

Formally, we construct a new set of positive samples $T_{V,P} = R_{V,P} \subseteq R_V$, where R_V is the set of visual search results,for all $s_i \in R_{V,P}, d(Q,i)_V < d_{V,l\text{thres}}$ and $d_{V,l\text{thres}}$ is an experimentally set threshold. Subsequently, we define a set of negative samples $T_{V,N} = R_{V,N} \subseteq R_V$, where for all $s_i \in R_{V,N}, d_V(Q,i) > d_{V,h\text{thres}}$. These distance thresholds could either experimentally set to specific values, where

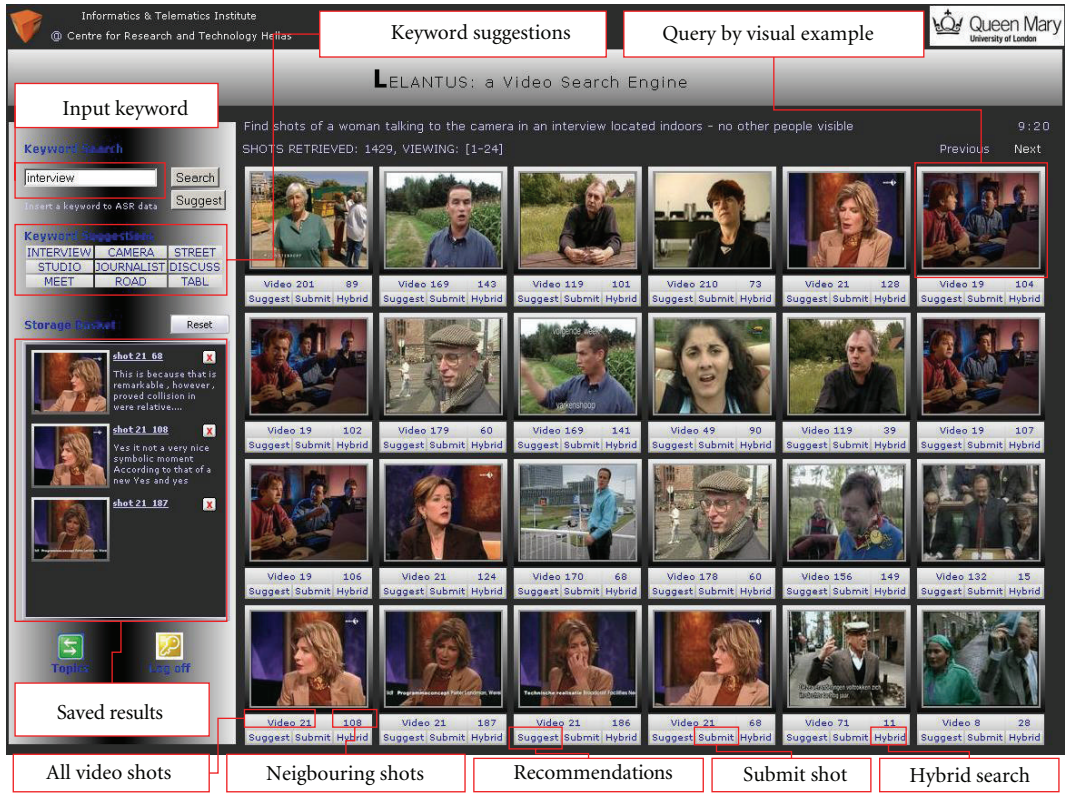

Figure 7: Search engine interface.

Figure 8: GUI showing the temporal neighboring shots and the associated ASR text of a shot.

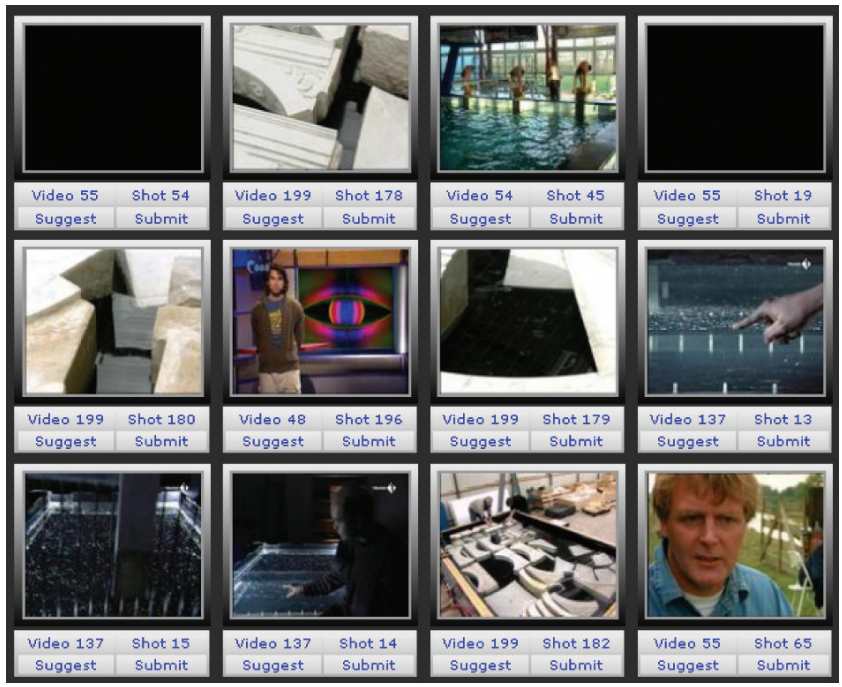

Figure 9: The user submits a textual query with the keyword "water" searching in the ASR transcripts.

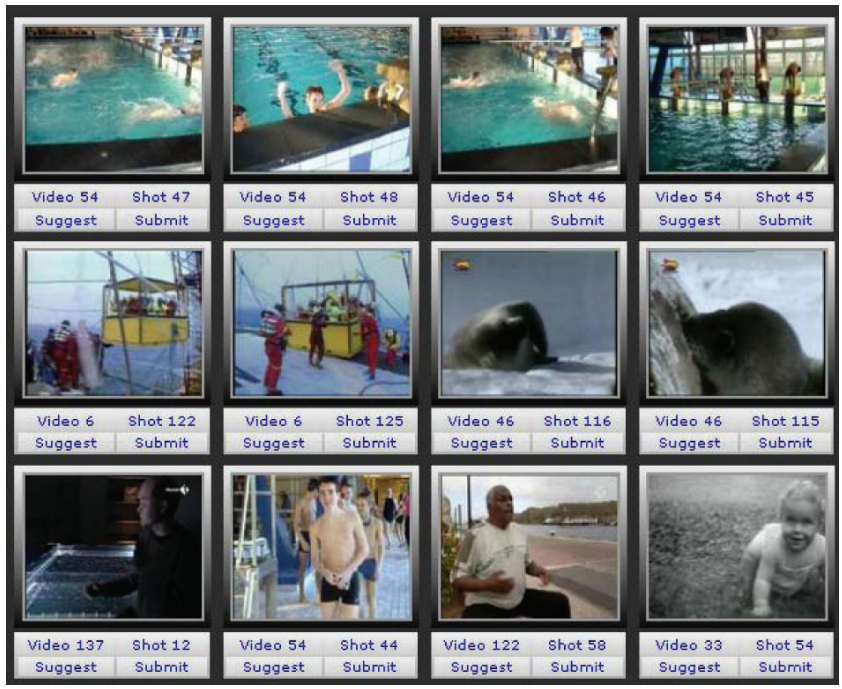

Figure 10: Results from a textual query (keyword "water") searching with the aid of the weighted graph.

always $d_{V,l\text{thres}} > d_{V,l\text{thres}}$ or they could be manually adjusted by the user in a manual assisted combination of implicit information and visual data according to the user needs. The final training set is

$$T = T_P \cup T_N = (T_{I,P} \cup T_{V,P}) \cup (T_{I,N} \cup T_{V,N}). \quad (6)$$

7.2. Support Vector Machine Classifier. As the training and the reranking of the results is performed in real time during the query, we have to select a fast implementation which can provide results in reasonable time. The advantage of performing the training on line is that in a semiautomatic version of the module, the user would be able to optimize the combination procedure by adjusting weights for the two

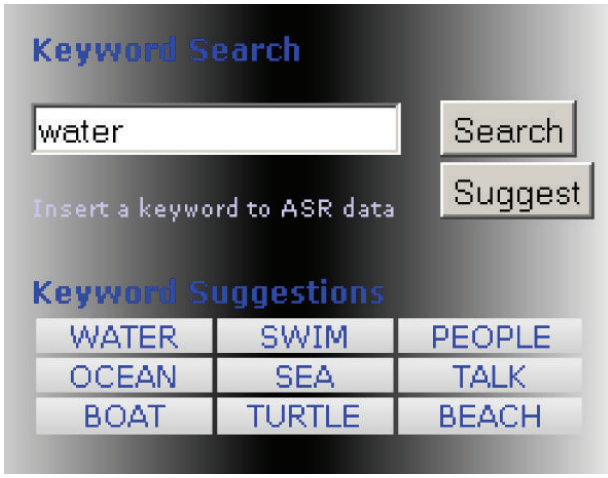

Figure 11: Keyword suggestions for a text-based query.

involved rankings (i.e., visual and implicit), which would reflect to the definition of different distance thresholds in the training data construction. Of course apart from the implementation, the size of the training dataset comprises an important factor that could keep the speed low. As our objective is to rerank results, the initial idea was to employ the ranking SVM of [11], which could be trained by user preferences between two different shots. However, based on how the weighted graph is constructed and especially the quantitive assignment of the actions' weights, it is clear that the shots that are closer to the example shots are better to be considered as positive samples instead of user relative preference declaration.

Hence, we employ the fast-performing SVM classifier implementation described in Section 3 utilizing as features of the positive and negative samples the concatenation of the edge histogram and color layout descriptors. Assuming that the concatenated visual descriptor is $V = \{v_0, v_1, \ldots, v_P\}$, then (2) is transformed into

$$\forall c \in \{0,1\}^n : \frac{1}{n} w^T \sum_{i=1}^{n} c_i y_i V_i \geq \frac{1}{n} \sum_{i=1}^{n} c_i - \xi. \quad (7)$$

After having trained the classifier with T, we provide as test set the initial results R_V based on visual descriptors. This test set is finally ranked based on the distances that are calculated between each shot and the hyperplane, which is constructed by the model.

8. LELANTUS Search Engine

In this section, we will present LELANTUS (LELANTUS in Greek Mythology was a Titan who had the capability of "moving without being seen" implying that the engine collects and processes the implicit user feedback in a transparent way to the user) video search engine which realizes the proposed framework. First, we will describe the interface, and then we will demonstrate its functionalities through user interaction modes. An interactive demonstration of LELANTUS video search engine is available at: http://mklab-services.iti.gr/lelantus/.

8.1. Interface. The search engine interface is illustrated in Figure 7. All the highlighted functionalities can be recorded during the user interaction. Taking a closer look, we observe that the graphical user interface (GUI) is composed of two parts: the left column, which offers text-based search options, and the main container, where the results are presented offering at the same time options for queries by visual example and temporal search.

At the left column the user is allowed to enter a keyword in order to fire a text-based search. Two different options are offered: (i) to perform a textual search exploiting the ASR information and (ii) to search based on the weighted graph and receive recommendations. Below this part, a number of related keywords are presented for every query submission using the weighted graph. Finally, the left column includes a basket storage structure, where the user can store the results he/she finds.

The main container is the part where the results are presented. Six different options are available for each shot: (i) to perform a query by visual example using the MPEG descriptors by clicking on the representative image, (ii) to mark a shot as relevant to the topic (i.e., submit a shot), (iii) to view all the shots of the same video, (iv) to fire an image by example search using the relations of the user interaction weighted graph, and (v) to execute a hybrid search, which combines visual features and implicit user feedback and finally as discussed in Section 7 and (vi) to view the temporally adjacent (i.e., neighboring) shots of a selected video shot with the associated textual transcription, as it is shown in Figure 8.

8.2. Interaction Modes. In this section, the improvement of results, when implicit feedback is taken into account is demonstrated by presenting different interaction modes. In these examples, the system is accessing the TRECVID 2008 test data collection (presented in detail in Section 9). The implicit user feedback information has been gathered during the first experimental phase described in Section 9.2.

First, we present a usage scenario, in which the user is searching for scenes where a water body is visible by typing the keyword "water" (Figure 9). As text retrieval is performed on the noisy information provided by automatic speech recognition (ASR), only some of the results depict water scenes. Conducting the same query utilizing the graph with the past interaction data (i.e., the recommendations), we get a clearly better set of results (Figure 10). At the same time, the system outputs term recommendations using the weighted graph (Figure 11). In this case, the query keyword was "water" and most of the recommended words seem to have high semantic similarity with the input term.

In the second usage scenario, the user is employing the query by visual example methodology to find images that are similar to a given one. Subsequently, we present the set of results when the user searches with the three different

Figure 12: Query by image example. Content-based analysis is employed for this query. The input image is the one on the left top corner.

Figure 13: Query by image example. Relations from the weighted graph are used to realize the query. The input image is the one on the left top corner.

modalities: (a) content-based search using the visual features, (b) graph-based recommendations utilizing the past user interaction, and (c) hybrid search, which combines visual features and implicit user feedback. The output of the system is visualized in Figures 12, 13, and 14, respectively, for the aforementioned search flavors, while the related terms generated by the system for this specific query are illustrated in Figure 15. If we attempt to comment on the visual output of the system, it seems that the top 12 results of the hybrid approach are of better quality compared to the results of the other search modules. More specifically, a precision of 91.67% (11/12) is reported for hybrid modality,

Figure 14: Hybrid search combining visual features and implicit user feedback. The input image is the one on the left top corner.

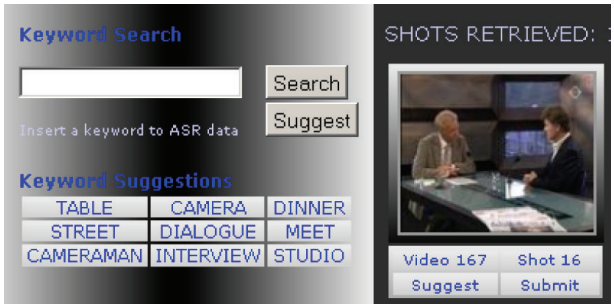

Figure 15: Keyword suggestions in a query by image example. The input image is the one at the right.

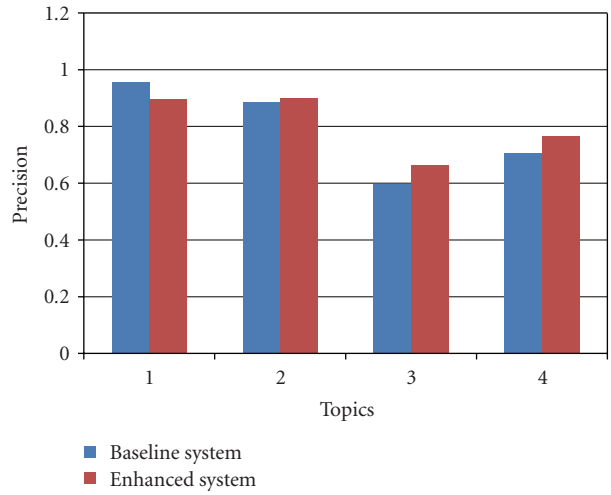

Figure 16: Precision for the results of the baseline and enhanced systems.

while the visual only based search achieves a precision of 58.3% (7/12). Of course, the good performance of the hybrid approach is due to the high precision 75% (9/12) reported by the recommendations (Figure 13), as the latter are actually used for the construction of the training set described in Section 7.1.

In this specific case, it seems that when retrieval is performed only by considering the visual descriptors, low distances are estimated also for shots that have similar colors but no semantic resemblance with the query. On the other hand, when the hybrid ranking is applied, it seems that the implicit user feedback has given a semantic flavor to the ranking as the shots that shared only common color characteristics were ranked lower. As far as the term recommendations are concerned, most of the suggested words (i.e., 7 of 9) seem to be semantically related to the selected image (Figure 15).

Table 2: Numerical statistics for the weighted graph.

	Nodes	Shots	Keywords	Links
Weighted graph	1298	1229	69	2659

9. Experiments and Evaluation

In order to evaluate the proposed approach, we employed the LELANTUS video search engine to conduct experiments and make comparisons between the different retrieval modalities. The evaluation experiment was divided into 3 phases: the

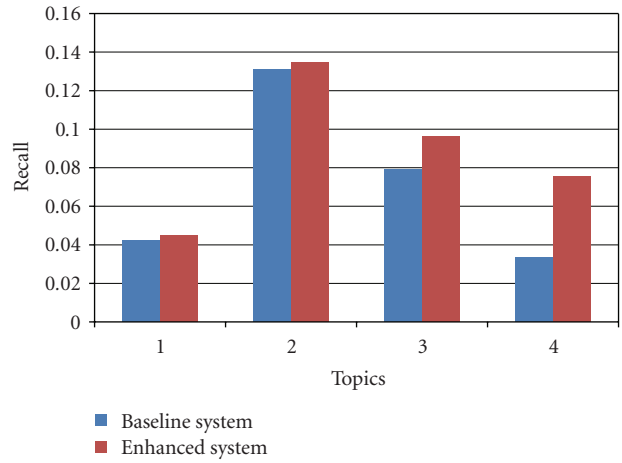

Figure 17: Recall for the results of the baseline and enhanced systems.

Table 3: Precision and Recall for the Baseline and Enhanced systems.

Topics	Precision		Recall	
	Baseline	Enhanced	Baseline	Enhanced
1	0.955556	0.896296	0.042042	0.045045
2	0.885467	0.900362	0.130719	0.134804
3	0.597154	0.664683	0.07906	0.096154
4	0.707702	0.766934	0.033712	0.075335

training phase, in which the implicit user feedback was recorded, the evaluation of the generated recommendations, and finally the evaluation of the hybrid search modality.

9.1. Dataset Involved and Ground Truth. In these experiments we made use of the annotated video set of TRECVID 2008. This set includes about 100 hours of Dutch video (news magazine, science news, news reports, documentaries, educational programming, and archival video). The videos are segmented into about 30.000 shots (shot segmentation offered by NIST) and are presented by the most representative keyframe in the search engine. The ASR is offered by the University of Twente (http://www.utwente.nl/); however, due to the errors that are usually employed during automatic speech recognition and machine translation from Dutch to English, they cannot be considered as highly reliable. The ground truth is provided by the National Institute of Standards and Technology (NIST: http://www.nist.gov/) and included annotated shots for 24 query topics. Part of these query topics were used in our experiments.

9.2. Training Phase. In the first phase (i.e., the training phase), 24 search sessions, each lasting 15 minutes, took place, in which 24 users searched for the following 6 different topics (i.e., 4 different users for each topic) and their actions were recorded.

(A) Find shots of 3 or fewer people sitting at a table.

(B) Find shots of one or more people with mostly trees and plants in the background; no road or building is visible.

(C) Find shots of one or more people where a body of water can be seen.

(D) Find shots of a woman talking to the camera in an interview located indoors—no other people visible.

(E) Find shots of one or more pieces of paper, each with writing, typing, or printing it, filling more than half of the frame area.

(F) Find shots of a person on the street, talking to the camera.

Considering the fact that the users were not aware of the mechanism employed for building the weighted graph and that they were instructed to search as they normally do (i.e., without considering a specific strategy), they introduced some noise by performing random and unpredictable jumps during the search session.

In this phase, the baseline version (i.e., recommendation generation and hybrid search were disabled) of LELANTUS was used. All the functionalities of Figure 7 that are supported by the baseline system were recorded during the user interaction. Then, we constructed the weighted graph based on the proposed methodology. The numerical statistics (e.g., number of nodes, links, etc.) of the weighted graph are reported in Table 2.

9.3. Recommendations' Evaluation. In the second phase, we recruited 4 different users, who searched for the following 4 topics: 2 relevant (but not identical) to the ones of the first part and 2 irrelevant. In this case, each user searched for all the 4 topics.

(1) Find shots of one or more people with one or more horses.

(2) Find shots of a map (i.e., similar to topic (E)).

(3) Find shots of one or more people with one or more books.

(4) Find shots of food and/or drinks on a table (i.e., similar to topic (A)).

During these search sessions, the users used the baseline version of LELANTUS. Then, another 4 users performed a search for the same topics (1)–(4). These users were able to use not only the basic retrieval options of the system, but also the recommendation functionality (i.e., the enhanced version). The duration for each session was 10 minutes for the last two experimental phases.

Performance in terms of precision and recall for the 2nd and 3rd phases of the experiment is reported in Table 3 and illustrated in Figures 16 and 17, respectively, where these metrics are calculated against annotated results for each topic.

The average improvement in recall for the first two topics (1) and (2) (i.e., the irrelevant to the initial ones) is about

Figure 18: Combined ranking of results for the query shot on the top left corner.

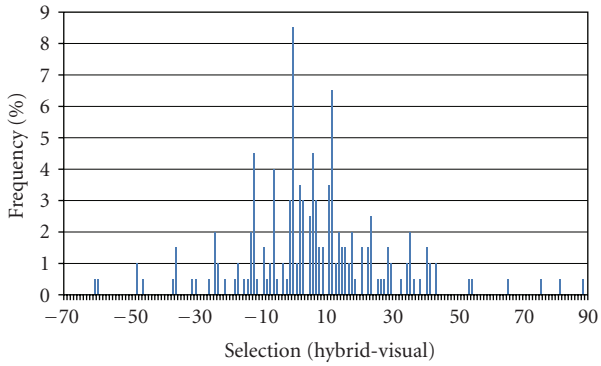

Figure 19: Histogram of the clicks. The horizontal axis stands for the number of clicks (positive for hybrid ranking and negative for visual), while the vertical for the frequency.

Table 4: Pairwise comparison of the hybrid retrieval function with the visual one.

Total queries	More selections on hybrid	More selections on visual	Equal selections
200	121	62	17

Table 5: Clicks on hybrid and visual ranking.

Total clicks	Selections on hybrid ranking	Selections on visual ranking
6509	5276	1233

5%, while precision seems to slightly drop by an average of 2%. These results can be considered rather reasonable due to the fact that past users have not interacted much with the video shots that are related to these topics. As expected, the major improvement is reported in the topics (3) and (4) (i.e., the relevant to the initial queries), in which recall and precision are increased by an average of 72% and 9.8%, respectively. The low absolute recall values are due to the fact that the shots that were annotated as relevant for each query-topic could not possibly be retrieved in the requested time duration of the experimental search sessions of this phase.

9.4. Visual Search Optimization Experiment. In this experimental phase, we attempt to evaluate the hybrid search modality that combines visual features and implicit user feedback.

To measure and evaluate the performance of the suggested algorithm, we compare the two different rankings in the case of a query by visual example: (a) the visual ranking, which is provided by the distances of the visual descriptors and (b) the hybrid ranking, which is generated after the application of the proposed algorithm. To compare the aforementioned retrieval functions, we utilize the method suggested in [11]. Considering that RK_H is the hybrid ranking and RK_V the visual, we construct a "combined ranking" $RK_{H,V}$ that includes the top links of both rankings. More specifically, $RK_{H,V}$ consists of l results, which are actually the k_a top results of RK_H and the k_b of RK_V, where $|k_a - k_b| \leq 1$. This form of presentation leads to a blind statistical test so that the user selections, expressed as clicks on the ranked results, demonstrate the unbiased preferences. Such method can be considered even more appropriate when applied in visual query by example instead of text web search, as in this case, the results are not so subjective to

what the user has in mind. Figures 12 and 14 illustrate a visual and a hybrid ranking, respectively, while in Figure 18, the combined ranking for the same query shot is depicted.

In this phase, 12 users were employed to identify visually similar results for 200 queries by visual example that were included in the weighted graph. To construct efficiently the training set, the thresholds have been experimentally selected, while 30 positive and 30 negative examples were extracted by the weighted graph for each query. In Tables 4 and 5, we can observe the statistic results for all the users. It can be seen that for 60.5% of the queries the hybrid ranking is preferred by the users, for 8.5% of queries the two functions seem to perform equally, while for the rest 31% of queries the visual ranking outperforms the hybrid. In Figure 19, we can see the corresponding histogram, which shows the frequency of user preference (i.e., selection clicks on hybrid ranking minus clicks on visual ranking) for the involved queries. We constructed the histogram by considering absolute values of the clicks and not normalized (i.e., divided by the total clicks in a query), as the actual number of clicks seems to be of more importance. For instance, in a case that a user clicks and selects only one item more from the one of the rankings, the conclusion should be that this ranking is with higher probability slightly better than the other. This can be reflected when considering the absolute difference of the clicks (e.g., $9 - 8 = 1$ and $1 - 0 = 1$), where the value 1 describes the user preference. However, if we normalize the metrics according to the total number of selections in a query, we could get misleading results (e.g., 11.1% and 100%, resp., for the previous example).

10. Conclusions

In this paper, we have introduced new implicit interest indicators for video search and proposed a novel methodology to construct a content similarity graph based on the implicit indicators of patterns of user interaction. In addition, we have proposed an approach for combining effectively visual features with implicit user feedback by employing a SVM classifier. As it is shown by the results, the implicit user feedback can be of added value in video retrieval engines, considering also that large quantities of past user interaction data can easily become available. From the experimental results, it seems that the utilization of implicit feedback is capable of improving the visual search results in most of the cases, and in addition, it improves the system's performance. We could say that utilizing implicit user feedback to optimize visual search seems to tune the visual function in a semantic way, in which results with the same semantic concept are ranked higher despite the initial lower visual resemblance.

The approach is considered scalable due to the fact that it is based on well established structures as graphs, for which many efficient indexing techniques have been proposed. In addition, the action and weighted graphs are constricted offline and are updated in the background, when new user implicit feedback is gathered. Therefore, the application of the approach and especially the real time classification task (i.e., hybrid search) remain possible even in case of huge corpus.

Future work would include a more extended evaluation of the method, in order to better observe the behavior of this hybrid approach especially in cases that even more users are involved and the implicit feedback is of lower quality (e.g., in even less controlled experiments). Furthermore, the application of classification algorithms other than SVM, for example, boosting-based techniques and comparison between them would be of interest, while the employment of alternative visual features (e.g., SIFT) could also be considered. Finally, an important challenge would be to include information from the physical reactions of the user (e.g., eye movements) and combine it with user navigation patterns and video analysis information. Recently, our research activities are directed towards these goals.

Acknowledgment

This work was supported by the projects PESCaDO (FP7-248594) and PetaMedia (FP7-216444) both funded by the European Commission.

References

[1] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain, "Content-based multimedia information retrieval: state of the art and challenges," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 2, no. 1, pp. 1–19, 2006.

[2] N. Sebe, M. S. Lew, X. Zhou, T. S. Huang, and E. M. Bakker, "The state of the art in image and video retrieval," in *Proceedings of the 2nd International Conference on Image and Video Retrieval*, pp. 1–8, Urbana, Ill, USA, July 2003.

[3] M. L. Kherfi, D. Brahmi, and D. Ziou, "Combining visual features with semantics for a more effective image retrieval," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR '04)*, vol. 2, pp. 961–964, Cambridge, UK, August 2004.

[4] S. F. Chang, R. Manmatha, and T. S. Chua, "Combining text and audio-visual features in video indexing," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, vol. 5, pp. V1005–V1008, Philadelphia, Pa, USA, March 2005.

[5] S. Vrochidis, C. Doulaverakis, A. Gounaris, E. Nidelkou, L. Makris, and I. Kompatsiaris, "A hybrid ontology and visual-based retrieval model for cultural heritage multimedia collections," *International Journal of Metadata, Semantics and Ontologies*, vol. 3, no. 3, pp. 167–182, 2008.

[6] C. G. M. Snoek and M. Worring, "Multimodal video indexing: a review of the state-of-the-art," *Multimedia Tools and Applications*, vol. 25, no. 1, pp. 5–35, 2005.

[7] F. Hopfgartner, D. Vallet, M. Halvey, and J. Jose, "Search trails using user feedback to improve video search," in *Proceedings of the 16th ACM International Conference on Multimedia (MM '08)*, pp. 339–348, Vancouver, Canada, October 2008.

[8] X. S. Zhou, Y. Wu, I. Cohen, and T. S. Huang, "Relevance feedback in content-based image and video retrieval," in *Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Services*, pp. 1–12, Queen Mary University of London, 2003.

[9] G. Giacinto and F. Roli, "Instance-based relevance feedback for image retrieval," in *Advances in Neural Information Processing Systems*, L. K. Saul, Y. Weiss, and L. Bottou, Eds., vol. 17, pp. 489–496, MIT Press, 2005.

[10] C. Gurrin, D. Johansen, and A. F. Smeaton, "Supporting relevance feedback in video search," in *Proceedings of the 28th European Conference on IR Research (ECIR '06)*, pp. 561–564, London, UK, 2006.

[11] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*, pp. 133–142, Alberta, Canada, July 2002.

[12] F. Radlinski and T. Joachims, "Query chains: learning to rank from implicit feedback," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 239–248, Chicago, Ill, USA, August 2005.

[13] D. M. Nichols, "Implicit ratings and filtering," in *Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering*, pp. 31–36, Budapest, Hungary, November 1997.

[14] B. Yang, T. Mei, X. S. Hua, L. Yang, S. Q. Yang, and M. Li, "Online video recommendation based on multimodal fusion and relevance feedback," in *Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR '07)*, pp. 73–80, Amesterdam, The Netherland, July 2007.

[15] S. Vrochidis, I. Kompatsiaris, and I. Patras, "Optimizing visual search with implicit user feedback in interactive video retrieval," in *Proceedings of the ACM International Conference on Image and Video Retrieval I (CIVR '10)*, pp. 274–281, July 2010.

[16] S. Vrochidis, I. Kompatsiaris, and I. Patras, "Exploiting implicit user feedback in interactive video retrieval," in *Proceedings of the 11th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS '10)*, Desenzano del Garda, Italy, April 2010.

[17] Y. Zhang, H. Fu, Z. Liang, Z. Chi, and D. Feng, "Eye movement as an interaction mechanism for relevance feedback in a content-based image retrieval system," in *Proceedings of the Eye Tracking Research and Applications Symposium (ETRA '10)*, pp. 37–40, Austin, Tex, USA, 2010.

[18] A. Yazdani, J. S. Lee, and T. Ebrahimi, "Implicit emotional tagging of multimedia using EEG signals and brain computer interface," in *Proceedings of SIGMM Workshop on Social Media*, pp. 81–88, New York, NY, USA, 2009.

[19] D. Kelly and J. Teevan, "Implicit feedback for inferring user preference: a bibliography," *SIGIR Forum*, vol. 32, no. 2, 2003.

[20] M. Claypool, P. Le, M. Wased, and D. Brown, "Implicit interest indicators," in *Proceedings of the International Conference on Intelligent User Interfaces (IUI '01)*, pp. 33–40, Santa Fe, Mex, USA, 2001.

[21] R. White, I. Ruthven, and J. M. Jose, "The use of implicit evidence for relevance feedback in web retrieval," in *Proceedings of the 24th BCS-IRSG European Colloquium on IR Research: Advances in Information Retrieval*, pp. 93–109, Glasgow, UK, 2002.

[22] F. Hopfgartner and J. Jose, "Evaluating the implicit feedback models for adaptive video retrieval," in *Proceedings of the International on Multimedia Information Retrieval*, pp. 323–331, Bavaria, Germany, 2007.

[23] D. Vallet, F. Hopfgartner, and J. Jose, "Use of implicit graph for recommending relevant videos: a simulated evaluation," in *Proceedings of the 30th Annual European Conference on Information Retrieval (ECIR '08)*, vol. 4956 of *Lecture Notes in Computer Science*, pp. 199–210, Glasgow, Scotland, 2008.

[24] T. Joachims, "Training linear SVMs in linear time," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*, pp. 217–226, Philadelphia, Pa, USA, August 2006.

[25] T. Joachims, "A support vector method for multivariate performance measures," in *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pp. 377–384, Bonn, Germany, August 2005.

[26] A. Hanjalic, R. L. Lagendijk, and J. Biemond, "A new method for key frame based video content representation," in *Image Databases and Multimedia Search*, A. Smeulders and R. Jain, Eds., pp. 97–107, World Scientific, 1997.

[27] S. U. Naci, U. Damnjanovic, B. Mansencal et al., "The COST292 experimental framework for rushes summarization task," in *Proceedings of TRECVID Workshop*, pp. 40–44, Gaithersburg, Md, USA, 2008.

[28] Kinosearch Search Engine Library, http://www.rectangular.com/kinosearch/.

[29] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, pp. 130–137, 1980.

[30] S. E. Robertson and K. S. Jones, "Simple, proven approaches to text retrieval," Tech. Rep. UCAM-CL-TR-356, University of Cambridge, Computer Laboratory, Cambridge, UK, 1994.

[31] MPEG-7 XM Software, http://www.lis.ei.tum.de/research/bv/topics/mmdb/e_mpeg7.html.

[32] A. Guttman, "R-trees: a dynamic index structure for spatial searching," in *Proceedings of the ACM International Conference on Management and Data (SIGMOD '84)*, pp. 47–57, New York, NY, USA, 1984.

[33] S. Vrochidis, P. King, L. Makris et al., "MKLab interactive video retrieval system," in *Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR '08)*, pp. 563–563, Niagara Falls, Canada, 2008.

[34] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[35] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.