Video Retrieval from Few Examples Using Ontology and Rough Set Theory

Kimiaki Shirahama[1] and Kuniaki Uehara[2]

1. Graduate School of Economics, Kobe University,
2-1, Rokkodai, Nada, Kobe, 657-8501, Japan
2. Graduate School of System Informatics, Kobe University,
1-1, Rokkodai, Nada, Kobe, 657-8501, Japan
shirahama@econ.kobe-u.ac.jp, uehara@kobe-u.ac.jp

Abstract. In query-by-example approach, a user can only provide a small number of example shots to represent a query. In contrast, depending on camera techniques and setting, relevant shots to the query are characterized by significantly different features. Thus, we develop a video retrieval method which can retrieve a large variety of relevant shots only from a small number of example shots. But, it is difficult to build an accurate retrieval model only from a small number of example shots. Consequently, the retrieval result includes many shots which are clearly irrelevant to the query. So, we construct an ontology as a knowledge base for incorporating object recognition results into our method. Our ontology is used to select concepts related to the query. By referring to recognition results of objects corresponding to selected concepts, we filter out clearly irrelevant shots. In addition, we estimate a parameter of a retrieval model based on the correlation between selected concepts and shots retrieved by the model. Furthermore, to retrieve a variety relevant shots characterized by different features, we use "rough set theory" to extract multiple classification rules for identifying example shots. That is, each rule is specialized to retrieve relevant shots characterized by certain features. Experimental results on TRECVID 2009 video data validate the effectiveness of our method.

1 Introduction

Recently, there is a great demand to develop a video retrieval method, which can efficiently retrieve interesting shots from a large amount of videos. In this paper, we develop a method based on "Query-By-Example (QBE)" approach, where a user represents a query by providing example shots. Then, QBE retrieves shots similar to example shots in terms of color, edge, motion, and so on. We consider QBE as very effective, because a query is represented by features in example shots without the ambiguity of semantic contents. In addition, QBE can retrieve any interesting shots as long as users can provide example shots.

However, QBE is challenging because in shots with similar features, semantic contents are not necessarily similar to each other. For example, when *Ex. 1* in

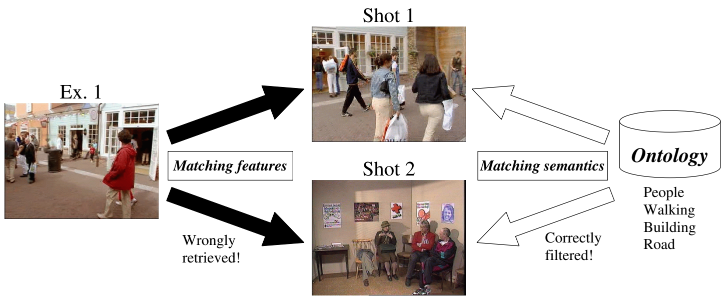


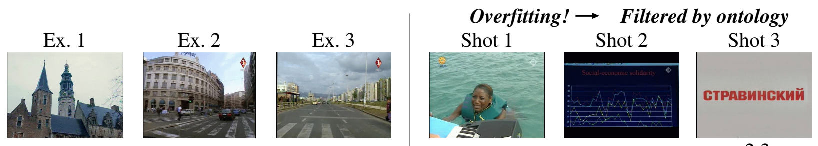
Fig. 1. Example of QBE using an ontology for the query "people walk in a street".

Fig. 1 is provided as an example shot for the query "people walk in a street", in addition to *Shot 1*, *Shot 2* is wrongly retrieved. The reason is that both of *Ex. 1* and *Shot 2* have red-colored and ocher-colored regions. Also, instead of rectangular buildings and windows in *Ex. 1*, rectangular posters are put up on the wall in *Shot 2*. Like this, one drawback of QBE is that it ignores semantic contents. Thus, we incorporate an ontology as a knowledge base into QBE.

We consider that an ontology is especially important for alleviating a lack of example shots in QBE. Usually, a user can only provide a small number of example shots (at most 10). In contrast, we represent each shot by using very high-dimensional features. For example, a popular SIFT feature leads to a shot representation with more than 1000 dimensions, where each dimension represents the frequency of a local edge shape (i.e. visual word). Generally, as the number of feature dimensions increases, the number of example shots needed to construct a well generalized retrieval model exponentially increases [1]. This means that the statistical information of features in a small number of example shots is not reliable. So, a retrieval model tends to be overfit to feature dimensions which are very specific to example shots but ineffective for characterizing relevant shots. For example, in Fig. 2, if *Ex. 1*, *Ex. 2* and *Ex. 3* are provided as example shots, the retrieval model is overfit to feature dimensions which characterize a small number of edges in the upper part (i.e. sky regions). As a result, it retrieves *Shot 1*, *Shot 2* and *Shot 3* which are clearly irrelevant to the query.

In order to filter out clearly irrelevant shots, we develop an ontology for utilizing object recognition results. Fig. 2 shows recognition results for three objects, *Building*, *Cityspace* and *Person*. Here, one shot is represented as a vector of recognition scores, each of which represents the presence of an object. In Fig. 2, we can see that *Building* and *Cityspace* are likely to appear in example shots while they are unlikely to appear in the other shots. Recently, researchers use object recognition results in video retrieval [1]. For example, research groups in City

[1] Objects are frequently called "concepts" in the filed of video retrieval. But, some readers may confuse them with concepts which are hierarchically organized in an ontology. So, we use the term "concept" only when it constitutes an ontology.



	Ex. 1	Ex. 2	Ex. 3	Shot 1	Shot 2	Shot 3
Building:	2.5	2.2	1.2	-0.3	-1.0	-2.3
Cityspace:	1.1	2.6	1.5	-0.5	-1.5	-1.2
Person:	-1.0	-0.5	0.2	2.0	-1.3	-0.8

Fig. 2. An example of an overfit retrieval result for the event "tall buildings are shown".

University of Hong Kong [3] and University of Amsterdam [2] build classifiers for recognizing 374 and 64 objects, respectively. In particular, such classifiers are built by using a large amount of training data (e.g. $61,901$ shots in [3] and more than $10,000$ shots in [2]). Thereby, objects can be robustly recognized independently of sizes, positions and directions on the screen. The effectiveness of using object recognition results is proved in TRECVID, which is a famous annual international workshop on video retrieval [4].

To utilize object recognition results, we port objects into concepts in an ontology. Specifically, we define a hierarchical structure of concepts and concept properties. Thereby, we can select concepts related to the query, and examine recognition scores of objects corresponding to selected concepts. For example, in Fig. 2, if *Building* and *Cityspace* are selected, *Shot 1*, *Shot 2* and *Shot 3* can be filtered out due to low recognition scores for *Building* and *Cityspace*. Also, filtering irrelevant shots reduces the computation time. Furthermore, we introduce a method for building an accurate retrieval model based on the correlation between concepts selected for a query and shots retrieved by the model. Note that we are not given the label of a shot (i.e. relevant or irrelevant), but given object recognition scores. Thus, we assume that an accurate retrieval model preferentially retrieves shots which have high recognition scores of objects, corresponding to concepts related to the query.

We address another important problem in QBE, where even for the same query, relevant shots are taken in many different shooting environments. As can be seen from example shots in Fig. 2, shapes of buildings and regions where they are shown are significantly different from each other. Additionally, in each of *Ex. 2* and *Ex. 3*, a road is shown while it is not shown in *Ex. 1*. So, shots relevant to the query are characterized by significantly different features. Regarding this, typical QBE methods only use one example and retrieve shots similar to it [5, 6]. As a result, many relevant shots are inevitably missed. Compared to this, we use multiple example shots and "Rough Set Theory (RST)" which is a set-theoretic classification method for extracting rough descriptions of a class from imprecise (or noisy) data [7]. By using RST, we can extract multiple classification rules which can correctly identify different subsets of example shots. Thereby, we can retrieve a variety of relevant shots where each classification rule is specialized to retrieve a portion of relevant shots characterized by certain features.

2 Related Works

2.1 Concept selection for Video Retrieval

The most popular ontology for video retrieval is "Large-Scale Concept Ontology for Multimedia (LSCOM) [8]". It targets at broadcast news videos and defines a standardized set of $1,000$ concepts. But, LSCOM just provides a list of concepts where no concept relation or structure is defined. So, many researchers explore how to appropriately select LSCOM concepts for a query.

Existing concept selection approaches can be roughly classified into three types, manual, text-based and visual-based selections. In manual concept selection, users manually select concepts related to a query [9]. But, different users select significantly different concepts for the same query. Specifically, [9] conducted an experiment where 12 subjects are asked to judge whether a concept is related to a query. As a result, only 13% of total $7,656$ judgements are the same among all subjects. In text-based concept selection, WordNet is frequently used where words in the text description of a query are expanded based on synonyms, hypernyms and hyponyms [2, 3]. Then, concepts corresponding to expanded words are selected. But, WordNet only defines lexical relations among concepts, and does not define spatial and temporal relations among concepts. For example, from WordNet, we cannot know that *Building* and *Road* are frequently shown in the same shots. Finally, in visual-based concept selection, concepts are selected as objects which are recognized in example shots with high recognition scores [2, 3]. But, this approach relies on accuracies of object recognition. LSCOM includes concepts corresponding to objects, which are difficult to be recognized, such as *Dogs*, *Telephone*, *Supermarket*, and so on. So, visual-based concept selection may wrongly select concepts which are unrelated to the query.

To overcome the above problems, we manually organize LSCOM concepts into an ontology, which can capture both lexical relations among concepts and their spatial and temporal relations. To do so, we define several new concepts which are missed in LSCOM. For example, we define a new concept *Air_Vehicle* as a superconcept of *Airplane* and *Helicopter*, in order to explicitly represent that both of *Airplane* and *Helicopter* fly in the air or move in airports. Also, we introduce a method which can appropriately estimate parameters of a retrieval model based on concepts selected by our ontology. To our best knowledge, there are no existing parameter estimation methods based on ontologies.

2.2 Rough Set Theory

One of the biggest advantages of RST is that it can extract multiple classification rules without any assumption or parameter. Specifically, by combining features characterizing example shots based on the set theory, RST extracts classification rules as minimal sets of features, needed to correctly identify subsets of example shots. Compared to this, although a "Gaussian Mixture Model (GMM)" can extract multiple feature distributions of example shots, these shots are not necessarily distributed based on Gaussian distributions [18]. Also, the

"Genetic Algorithm" (GA) can be used to extract multiple classification rules, where sets of features useful for identifying example shots are searched based on the principles of biological evolution [21]. But, parameters in the GA, such as the number of chromosomes, the probability of crossover and the probability of mutation, cannot effectively determined with no a priori knowledge. Furthermore, decision tree learning methods and sequential covering methods can be used extract multiple rules, but several useful rules are not detected as these methods depend on the order of extracting rule [19].

In RST, it is very important to determine which features characterize example shots. In other words, we need to define the indiscernibility relation between two example shots with respect to features. A traditional RST can deal only with categorical features, where the indiscernibility relation can be easily determined by examining whether two example shots have the same value or not [7]. But, in our case, example shots are represented by non-categorical high-dimensional features. To apply RST to such features, we built a classifier on each feature, and define the indiscernibility relation by examining whether two examples are classified into the same class or not. Although this kind of classifier-based RST is proposed in [11], the high-dimensionality of features is not considered. Specifically, although [11] uses probabilistic classifiers such as naive bayes and maximum entropy, it is difficult to appropriately estimate probabilistic distributions only from a small number of example shots. Compared to this, we use Support Vector Machines (SVMs) which are known as effective for high-dimensional features [20]. [20] provides the theory that if the number of feature dimensions is large, SVM's generalization error is independent of the number of feature dimensions. In addition, even when only a small number of examples are available, the margin maximization needs no probability distribution estimation. Therefore, we develop a classifier-based RST using SVMs.

3 Video Retrieval Method

First of all, we set the condition where our QBE method is developed. We use large-scale video data provided by TRECVID [4]. This data consists of 219 development and 619 test videos in various genres, like cultural, news magazine, documentary and education programming. Each video is already divided into shots by using an automatic shot boundary detection method, where development and test videos include $36,106$ and $97,150$ shots, respectively. Like this, TRECVID video data is sufficient for evaluating the effectiveness of our QBE method on large-scale video data.

In order to filtering out clearly irrelevant shots to a query, we borrow recognition results of 374 objects, provided by the research group in City University of Hong Kong [3]. That is, recognition scores of 374 objects are associated with all shots in development and test videos. To utilize the above recognition results, we develop an ontology where LSCOM concepts corresponding to 374 objects are organized. Also, to extract features used in RST, we use the color descriptor software [13]. Specifically, we extract the following 6 different types of features from the middle video frame in each shot: *1. SIFT*, *2. Opponent SIFT*, *3. RBG SIFT*, *4. Hue SIFT*, *5. Color histogram* and *6. Dense SIFT* (see [13] in more detail). For each type of feature, we extract $1,000$ visual words by clustering $200,000$ features sampled from development videos. That is, we represent a shot as a total $6,000$-dimensional vector, where each type of feature is represented as a $1,000$-dimensional vector. Finally, for a query, we manually collect example shots from development videos, and retrieve relevant shots in test videos.

3.1 Building Ontology for Concept Selection

Fig. 3 shows a part of our ontology. LSCOM concepts are represented by captital letters followed by lower-case letters, while concepts that we define are represented only by capital letters. Also, we represent properties by starting their names with lower-case letters. Our ontology is developed by considering the "disjoint partition" requirement. This is a well-known ontology design pattern for making our ontology easily interpretable by both human and machine [14]. The disjoint partition means that a concept C_1 should be decomposed into disjoint subconcepts $C_2, C_3, \cdots$. That is, for $i, j \geq 2$ and $i \neq j$, $C_i \cap C_j = \phi$. So, an instance of C_1 cannot be an instance of more than one subconcept $C_2, C_3, \cdots$. For example, we should not place *Vehicle* and *Car* in the same level of the concept hierarchy, because an instance of *Car* is an instance of *Vehicle*. Thus, we have to carefully examine whether a concept is a generalization (or specialization) of another concept.

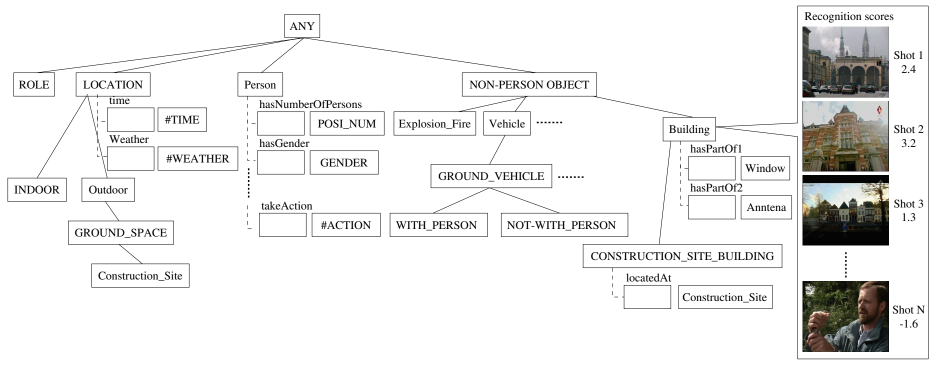


Fig. 3. A part of our ontology.

Furthremore, we consider visual characteristics to define our concept hierarchy. For example, as can be seen from Fig. 3, we define two subconcepts of *GROUND_VEHICLE*, *WITH_PERSON* and *NOT-WITH_PERSON*. We can induce that *Person* probably appears in shots containing subconcepts of *WITH_PERSON*, such as *Bicycle* and *Motorcycle*. On the other hand, it is uncertain that *Person* appears in shots containing subconcepts of *NOT-WITH_PERSON*.

Now, we explain how to select concepts related to a query. Basically, we firstly select concepts which match with words in the text description of the query. Then, for each selected concept, we select its subconcepts and concepts which are specified as properties. For example, for the query "buildings are shown", *Buildings* and all of its subcocepts (e.g. *Office_Buildings*, *Hotel*, *Power_Plant* etc.) are firstly selected. Then, as shown in Fig. 3, *Windows* and *Antenna* are selected from *hasPartOf1* and *hasPartOf2* properties of *Building*. After that, from *locatedAt* property of *CONSTRUCTION_SITE_BUILDING* (a subconcept of *Building*), we select *Construction_Site* and all of its subconcepts (e.g. *Cityspace*, *Urban*, *Suburban* etc.). At this point, by tracing concept properties many times, we may select concepts which are unrelated to the query. For example, from the above *Construction_Site*, we can trace *ARTIFICIAL_ROAD*, *Sidewalk* and *Person*. But, these concepts are not related to the query. To avoid selecting unrelated concepts, we restrict the number of tracing concept properties to only one time. That is, for the above example, we finish concept selection after selecting *Construction_Site* and all of its subconcepts.

In Fig. 3, some concept properties are characterized by slots where # precedes concept names. We call such an operator "# operator" which represents a concept property, used only when it is specified in the textual description of a query. Let us consider the query "people are indoors". For this query, we select *Person* and all of its subconcepts, and trace *Person*'s concept properties. But, for "takeAction" property, the current LSCOM only defines 12 concepts, such as *Singing*, *People_Crying*, *Talking* and so on. If these concepts are selected, shots containing them may be preferred. As a result, we may miss many shots where people take many other actions in indoor situations, such as eating and watching TV. Thus, only for queries like "people talking indoor", we use the concept property "takeAction" to select concepts.

Since the textual description of a query is usually simple, we cannot select concepts which are definitely related to the query. For example, for the query "buildings are shown", we select 55 concepts including *White_House*, *Military_Base*, *Ruins*, and so on. But, only a part of these concepts are truly related to the query. So, we validate selected concepts using example shots. Recall that all shots are associated with recognition scores of objects corresponding to LSCOM concepts, as shown in *Building* in Fig. 3. Based on such recognition scores in example shots, we validate concepts selected by our ontology. Specifically, for each object corresponding to a concept, we compute the average recognition score among example shots. Then, we rank concepts in the descending order. After that, we select concepts which are not only selected by our ontology, but also ranked in top T positions (we use $T = 20$). Like this, selected concepts are validated from both semantic and statistical perspectives.

Finally, we explain how to estimate classifier's parameter based on object recognition scores. Note that this classifier is an SVM used to define indiscernibility relations among example shots in RST. Suppose that for a query, we have a set of selected concepts C, where each concept is represented as c_i $(1 \leq i \leq |C|)$. Also, we have P parameter candidates for an SVM M, where the j-th parameter is p_j and the SVM with p_j is M_{p_j} $(1 \leq j \leq P)$. To estimate the best parameter, we temporarily retrieve S shots by using M_{p_j} (we use $S = 1,000$). Then, we compute the correlation between C and M_{p_j} as follows:

$$Correlation(C, M_{p_j}) = \sum_{i=1}^{C} \gamma(rank(M_{p_j}),\ rank(c_i)) \tag{1}$$

where $rank(M_{p_j})$ represents a ranking list of S shots according to their evaluation values by M_{p_j}. We obtain these evaluation values as SVM's probabilistic outputs [17]. $rank(c_i)$ represents a ranking list of S shots according to recognition scores of the object corresponding to c_i. We compute $\gamma(rank(M_{p_j}), rank(c_i))$ as the Spearman's rank correlation coefficient [15]. It represents the correlation between two ranking lists. If these are highly correlated, $\gamma(rank(M_{p_j}), rank(c_i))$ is close to 1, otherwise close to -1. So, a larger $\gamma(M_{p_j}, c_i)$ indicates that M_{p_j} is more correlated with c_i. $Correlation(C, M_{p_j})$ represents the overall correlation over all concepts in C. Thus, we select the best parameter p_j^* where $Correlation(C, M_{p_j})$ is the largest among P parameters. In this way, we can estimate an SVM parameter which is semantically validated based on selected concepts.

3.2 Video Retrieval Using Rough Set Theory

We use rough set theory (RST) to extract classification rules, called "decision rules", for discriminating relevant shots to a query from all irrelevant shots. To this end, we need two types of example shots. The first type of example shots are provided by a user and serve as representatives of relevant shots ("positive examples (p-examples)"). The second type of example shots serve as representatives of irrelevant shots ("negative examples (n-examples)"), but are not provided by the user. To overcome this, we have already developed a method which collects n-examples from shots other than p-examples [10]. Roughly speaking, our method iteratively enlarges n-examples by selecting shots which are more similar to already selected n-examples than p-examples. Thereby, our method can collect a variety of n-examples without wrongly selecting relevant shots as n-examples.

We discuss how to extract decision rules which can retrieve a large variety of relevant shots, only from a small number of p-examples. Note that decision rules are extracted by combining indiscernibility relations among examples, which are defined by SVMs. So, we need to build SVMs which can define various indiscernibility relations. To this end, we use "bagging" where SVMs are built on different sets of randomly sampled examples [16]. As described in [16], when only a small number of examples are available, SVMs' classification results are significantly different depending on examples. Thus, we can define various indiscernibility relations by building SVMs based on bagging. However, due to the high-dimensionality of features, SVMs may be overfit and may not appropriately define indiscernibility relations. To alleviate this, we use the "random subspace method" where SVMs are built on different sets of randomly sampled feature dimensions [12]. That is, the original high-dimensional feature is transformed into lower-dimensional features, so that we can alleviate to build overfit SVMs.

We regard SVM classification results as categorical features in RST, and extract decision rules for predicting the class of an unseen example. Let p_i and n_j be i-th p-example ($1 \leq i \leq M$) and j-th n-example ($1 \leq j \leq N$), respectively. a_k indicates the classification result by k-th SVM ($1 \leq k \leq K$). Here, $a_k(p_i)$ and $a_k(n_j)$ respectively represent class labels of p_i and n_j predicted by k-th SVM. That is, these are categorical features of p_i and n_j for a_k. In order to define the indiscernibility relation between each pair of p_i and n_j, RST extracts "discriminative features" which are useful for discriminating them. The set of discriminative features $f_{i,j}$ between p_i and n_j can be represented as follows:

$$f_{i,j} = \{a_k | a_k(p_i) \neq a_k(n_j)\} \tag{2}$$

That is, $f_{i,j}$ means that when at least one feature in $f_{i,j}$ is used, p_i can be discriminated from n_j.

Next, in order to discriminate p_i from all n-examples, we combine p_i's discriminative features. This is achieved by using at least one discriminative feature in $f_{i,j}$ for all n-examples. That is, we compute the following "discernibility function df_i" which takes a conjunction of $\vee f_{i,j}$:

$$df_i = \wedge\{\vee f_{i,j} |\ 1 \leq j \leq N\} \tag{3}$$

Let us consider the discernibility function df_1 for one p-example p_1 and two n-examples n_1 and n_2. Suppose that the set of discriminative features between p_1 and n_1 and the one between p_1 and n_2 are $f_{1,1} = \{a_1, a_3, a_5\}$ and $f_{1,2} = \{a_1, a_2\}$, respectively. Under this condition, df_1 is computed as $(a_1 \vee a_3 \vee a_5) \wedge (a_1 \vee a_2)$. This is simplified as $df_1^* = (a_1) \vee (a_2 \wedge a_3) \vee (a_2 \wedge a_5)$ [2]. That is, p_1 can be discriminated from n_1 and n_2, by using a_1, the set of a_2 and a_3 or the set of a_2 and a_5. Like this, each conjunction term in df_i^* represents a "reduct" which is a minimal set of features needed to discriminate p_i from all n-examples

From each reduct, we can construct a decision rule in the form of *IF-THEN* rule. Since each feature in our RST is defined by an SVM, a decision rule represents a combination of SVMs. For example, the decision rule constructed from the reduct $(a_2 \wedge a_3)$ is "*IF* a shot s is classified as positive by both 2-nd and 3-rd SVMs, *THEN* its class label is positive". That is, to match a decision rule with s, we examine whether s is classified as positive by all SVMs in the decision rule. In this way, we count how many decision rules match with s. Finally, we rank all shots in the descending order, and retrieve shots within top T positions (we use $T = 1,000$).

4 Experimental Results

We evaluate our method on the following 4 queries, *Query 1:* A view of one or more tall buildings and the top story visible, *Query 2:* Something burning with

[2] This simplification is achieved by using the distributive law $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$ and the absorption law $A \vee (A \wedge B) = A$.

flames visible, *Query 3:* One or more people, each at a table or desk with a computer visible, *Query 4:* One or more people, each sitting in a chair, talking. For each query, we run our method 9 times by using different sets of 10 p-examples. We evaluate the retrieval performance as the average number of relevant shots within 1,000 retrieved shots.

In Fig. 4 (a), we compare the following three types of retrieval, in order to evaluate the effectiveness of our ontology for filtering out irrelevant shots and estimating an SVM parameter. The first one is *Baseline* without using our ontology. The second type of retrieval is *Ontology1* which uses our ontology only for filtering out irrelevant shots. The final type of retireval is *Ontology2* which uses our ontology for both irrelevant shot filtering and SVM parameter estimation. For each topic, performances of *Baseline*, *Ontology1* and *Ontology2* are represented by the leftmost red bar, the middle green bar and the rightmost blue bar, respectively.

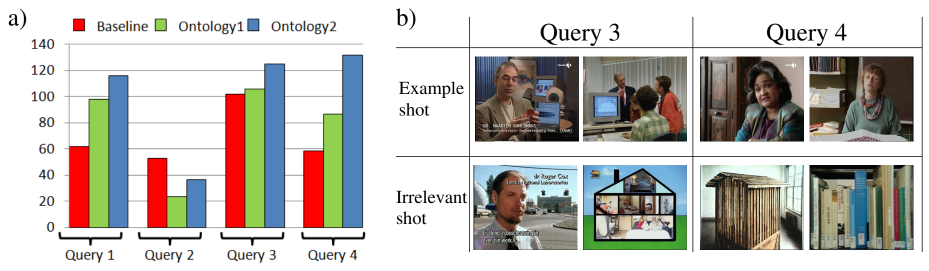


Fig. 4. (a) Performance comparison among *Baseline*, *Ontology1* and *Ontology2*, (b) Examples of shots filtered by our ontology.

As can be seen from Fig. 4 (a), except for *Query 2*, it is very effective to filter out irrelevant shots based on concepts selected by our ontology. The retrieval performance is further improved by estimating SVM parameters based on selected concepts. The reason for the low performance for *Query 2* is that, in each of 9 times retrieval, we can only select one or two concepts, that is, *Explosion_Fire* and *Earthquake*. What is worse, these concepts are not so effective for characterizing relevant shots. For example, 1,000 shots with highest recognition scores of *Explosion_Fire* and those of *Earthquake*, only characterize 37 and 12 relevant shots, respectively. As a result, we cannot appropriately filter out irrelevant shots and cannot appropriately estimate SVM parameters. To improve the performance for *Query 2*, other than *Explosion_Fire*, we need to recognize objects such as candle flame, bonfire, fire blasted from rockets, and so on.

In Fig. 4 (b), for each of *Query 3* and *Query 4*, we show two example shots and two clearly irrelevant shots which are filtered out by our ontology (for *Query 1*, see Fig. 2). For *Query 3*, *Baseline* without using our ontology wrongly retrieves shots where people just appear and shots which contains straight lines corresponding to computer shapes, and shapes of pillars and blinds in a background. For *Query 4*, *Baseline* wrongly retrieves shots which contain straight lines corresponding to shapes of background objects. By filtering out the above kind of shots, *Ontology1* and *Ontology2* can significantly outperform *Baseline*.

Finally, we examine whether filtering out irrelevant shots based ontology can reduce computation times. In Fig. 5, we show the average computation time of *Baseline* (left red bar) and *Ontology1* (right green bar) among of 9 times retrieval. As can be seen from this figure, filtering of irrelevant shots is useful for reducing computation times. Nonetheless, our method currently takes about 500 seconds, because it requires building of multiple SVMs and matching of many decision rules. So, from the perspective of computation cost, our method need to be improved. To this end, we are currently parallelizing processes of SVM building and decision rule matching by using multiple processors.

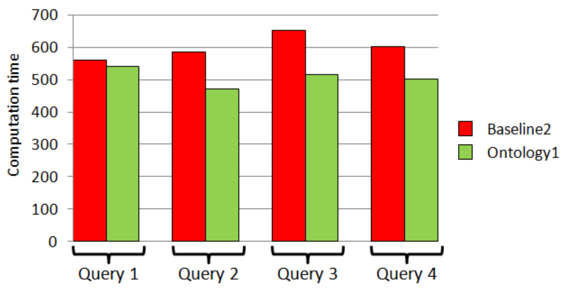


Fig. 5. Comparison between the computation time of *Baseline* and that of *Ontology1*.

5 Conclusion and Future Works

In this paper, we proposed a video retrieval method based on QBE approach. We construct an ontology to incorporate object recognition results into QBE. Specifically, our ontology is used to select concepts related to a query. By referring to recognition results of objects corresponding to selected concepts, we filter out clearly irrelevant shots. In addition, we estimate an SVM parameter based on the correlation between selected concepts and shots retrieved by the SVM. Also, to retrieve a large variety of relevant shots, we use RST for extracting multiple classification rules which characterize different subsets of example shots. Experimental results on TRECVID 2009 video data show that the retrieval performance can be significantly improved by using our ontology. Besides, our ontology is useful for reducing the computation time. Finally, by using RST, we can successfully cover a large variety of relevant shots.

Acknowledgments. This research is supported in part by Strategic Information and Communications R&D Promotion Programme (SCOPE) by the Ministry of Internal Affairs and Communications, Japan.

References

1. A. Jain, R. Duin and J. Mao: “Statistical Pattern Recognition: A Review”, TPAMI, Vol. 22, No. 1, pp. 4 – 37 (2000)
2. C. Snoek et al.: “The MediaMill TRECVID 2009 Semantic Video Search Engine”, In Proc. of TRECVID 2009, pp. 226 – 238 (2009)
3. C. Ngo et al.: “VIREO/DVM at TRECVID 2009: High-Level Feature Extraction, Automatic Video Search, and Content-Based Copy Detection”, In Proc. of TRECVID 2009, pp. 415 – 432 (2009)
4. A. Smeaton, P. Over and W. Kraaij: “Evaluation campaigns and TRECVid”, In Proc. of MIR 2006, pp. 321 – 330 (2006)
5. Y. Peng and C. Ngo: “EMD-Based Video Clip Retrieval by Many-to-Many Matching”, In Proc. of CIVR 2005, pp. 71 – 81 (2005)
6. K. Kashino, T. Kurozumi and H. Murase: “A Quick Search Method for Audio and Video Signals based on Histogram Pruning”, TMM, Vol. 5, No. 3, pp. 348 – 357 (2003)
7. J. Komorowski, A. Øhrn and A. Skowron: “The ROSETTA Rough Set Software System”, Handbook of Data Mining and Knowledge Discovery, W. Klösgen and J. Zytkow (eds.), chap. D.2.3, Oxford University Press (2002)
8. M. Naphade et al.: “Large-Scale Concept Ontology for Multimedia”, IEEE Multimedia, Vol. 13, No. 3, pp. 86 – 91 (2006)
9. M. Christel and A. Hauptmann: “ The Use and Utility of High-Level Semantic Features in Video Retrieval”, In Proc. of CIVR 2005, pp. 134 – 144 (2005)
10. K. Shirahama, C. Sugihara, Y. Matsuoka, K. Matsumura and K. Uehara: “Kobe University at TRECVID 2009 Search Task”, In Proc. of TRECVID 2009 Workshop, pp. 76 – 84 (2009)
11. S. Saha, C. Murthy and S. Pal: “Rough Set Based Ensemble Classifier for Web Page Classification”, Fundamenta Informaticae, Vol. 76, No. 1-2, pp. 171 – 187 (2007)
12. T. Ho: “The Random Subspace Method for Constructing Decision Forests”, TPAMI, Vol. 20, No. 8, pp. 832 – 844 (1998)
13. K. Sande, T. Gevers and C. Snoek: “Evaluating Color Descriptors for Object and Scene Recognition”, TPAMI, Vol. 32, No. 9, pp. 1582 – 1596 (2010)
14. M. Horridge et al.: “A Practical Cuide to Building OWL Ontologies Using The Prot*ég*é-OWL Plugin and CO-ODE Tools Edition 1.0”, http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf (2004)
15. R. Hogg and A. Crig: “Introduction to Mathematical Statistics (5th edition)”, Prentice Hall (1994)
16. D. Tao, X. Tang, X. Li and X. Wu: “Asymmetric Bagging and Random Subspace for Support Vector Machines-based Relevance Feedback in Image Retrieval”, TPAMI, Vol. 28, No. 7, pp. 1088 – 1099 (2006)
17. H. Lin, C. Jin and R. Weng: “A Note on Platt’s Probabilistic Outputs for Support Vector Machines”, Machine Learning, Vol. 68, No. 3, pp. 267 – 276 (2007)
18. J. Verbeek, N. Vlassis and B. Kröse: “Efficient Greedy Learning of Gaussian Mixture Models”, Neural Computation, Vol. 15, No. 2, pp. 469 – 485 (2003)
19. E. Alpaydin: “Introduction to Machine Learning”, MIT Press (2004)
20. V. Vapnik: “Statistical Learning Theory”, Wiley-Interscience (1998)
21. J. Han and M. Kamber: “Data Mining: Concepts and Techniques (Second Edition)”, Morgan Kaufmann Publishers (2006)