Kobe University at TRECVID 2011 Semantic Indexing and Multimedia Event Detection

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

Lin Yanpeng and Kuniaki Uehara
Graduate School of System Informatics, Kobe University
1-1, Rokkodai, Nada, Kobe 657-8501, Japan
lin@ai.cs.kobe-u.ac.jp, uehara@kobe-u.ac.jp

***Abstract*—This paper describes our methods and experimental results for TRECVID 2011 SIN and MED tasks. For SIN task, we submitted the run *L_A_cs24_kobe_sin_1* that addresses the following two problems: The first one is an expensive computation cost for constructing an SVM with a large number of examples. To ensure the detection accuracy and speed for each concept, we developed a method that selects a small number of negative examples similar to positive examples. Such negative examples are useful for characterizing the decision boundary between shots where the concept is present and shots where it is absent. The second problem is a large variety of shots where the concept is present, due to varied camera techniques and setting. Only using a single classifier is insufficient for covering such a large variety of shots. Hence, we used *rough set theory* to extract multiple classification rules, that characterize different subsets of positive examples. Although the evaluation result of *L_A_cs24_kobe_sin_1* was not very good, we found that one main reason is overfitting of classification rules extracted by RST.**

For MED task, we aim to examine the applicability of virtual reality techniques to event detection. It is laborious and time-consuming to collect a sufficient number of positive examples for an event. To overcome this, we create *virtual examples* using virtual reality techniques. We developed a method that creates virtual examples by synthesizing user's gesture, 3D object and background images. In order to evaluate the effectiveness of this approach, we submitted two runs, *c_real* (cs24-kobe_MED11_MED11TEST_MEDPart_SemiAutoEAG_c-real) and *p_virtual* (cs24-kobe_MED11_MED11TEST_MEDPart_SemiAutoEAG_p-virtual). *c_real* only uses positive examples that are selected from training videos. On the other hand, *p_virtual* uses virtual examples in addition to positive examples used in *c_real*. Detection error tradeoff curves of *c_real* and *p_virtual* indicate the effectiveness of virtual examples.

I. Introduction

This paper describes our methods and experimental results for Semantic INdexing (SIN) and Multimedia Event Detection (MED) tasks in TRECVID 2011. Our method for SIN task mainly addresses two problems:

Negative example selection: Through the collaborative annotation effort [1], a large number of positive and negative examples for a concept are available. They are used to construct a classifier that detects the presence or absence of the concept in shots. However, using all examples requires an expensive computation cost. In particular, our method determines the presence of the concept by combining outputs of many Support Vector Machines (SVMs) [2]. The computation cost of each SVM is $O(n^3)$ where n is the number of positive and negative examples [3]. Note that the classification of the SVM only requires support vectors that are the positive or negative examples closest to the decision boundary, and all other examples are redundant. Therefore, for fast SVM construction, we develop a Negative Example Selection (NES) method that can select a small number of negative examples, which are likely to become support vectors. Such negative examples should be visually similar to positive examples.

Large variation of shots: Features in shots where a concept is present significantly vary depending on camera techniques and settings. Thus, a single classifier is not capable of identifying such a large variety of shots. Hence, we use *Rough Set Theory* (RST) that is a set-theoretic classification method for extracting 'rough' descriptions of a class from imprecise (or noisy) data [4]. The term rough here indicates that RST does not extract a single classification rule to characterize the entire set of examples belonging to the class. Rather it extracts multiple rules that characterize different subsets of examples. The class is represented as a union of rules. We use RST to extract multiple rules, each of which characterizes a subset of positive examples. Such rules enables us to cover a variety of shots where the concept is present.

Our method for MED task is used to examine the applicability of *virtual reality techniques* to video event detection. It is laborious and time-consuming to collect a sufficient number of positive examples for an event. To overcome this, virtual reality techniques are used to create *virtual examples* where the event is presented by synthesizing user's gesture, 3D object and background images. We aim to evaluate the validity of substituting virtual examples with 'real' examples that are collected from training videos.

II. Semantic Indexing

Fig. 1 shows an overview of our SIN method. First, by applying the software developed by Sande et al. [5] to the middle video frame in each shot, we extract six types of

local image features, *SIFT*, *Dense SIFT*, *Opponent SIFT*, *RGB SIFT*, *Hue SIFT* and *RGB histogram*. These features are useful for characterizing a variety of shots where a concept is present, as they represent different color and edge properties in local image regions [5]. Each feature is represented using a $1,000$-dimensional bag-of-visual-words representation.

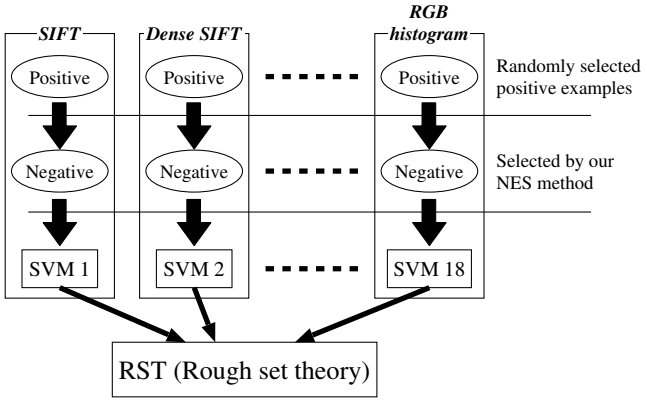

Figure 1. An overview of our semantic indexing method.

For each concept, our method first constructs many SVMs based on 'bagging'. As shown in Fig. 1, an SVM is constructed on one of the above features using randomly selected positive examples. Based on these, negative examples are selected our NES method. Specifically, for each feature, three SVMs are constructed using different sets of positive and negative examples. Such SVMs characterize a variety of shots where the concept is present, because of the difference in examples and features. However, these SVMs are not so accurate due to the insufficiency of positive and negative examples. Thus, RST is used to combine SVMs into classification rules, each of which can correctly discriminate a subset of positive examples from the whole of negative examples. Finally, the concept is regarded to be present in shots that match many rules. Below, we describe the details of our NES and RST methods.

A. Negative Example Selection

Our NES method aims to select a small number of negative examples that are similar to positive examples. These negative examples are likely to be support vectors, and useful for characterizing the boundary between shots where the concept is present and shots where it is absent. Algorithm 1 summarizes our NES method. For a concept, given a set of positive examples P and a set of negative examples N, our NES method outputs a shrunken set of negative examples similar to positive examples. As shown in line 1 to 5 in Algorithm 1, our NES method iteratively filters negative examples dissimilar to positive examples, using SVMs that are built on P and N consisting of the remaining negative examples. In what follows, this filtering is described in more detail.

Algorithm 1 An overview of our NES method

INPUT: Set of positive examples P, Set of negative examples N, Maximum number of iterations α, Ratio between negative examples and clusters β, Distance threshold $gamma$

OUTPUT: Shrunken set of negative examples N

$ite\_id = 0$

repeat

1. ite_id++
2. Cluster N into $|N|/\beta$ clusters
3. Obtain the set of representative negative examples RN, each of which is closest to a cluster center
4. Build an SVM using P and RN
5. From N, filter negative examples distant from the SVM's decision boundary using $gamma$

until $ite\_id \leq \alpha$ OR no negative example is filtered

return $N' = N$

We build an SVM on P and N where negative examples visually dissimilar to P are those distant from the decision boundary of the SVM. However, using all negative examples would impose a prohibitive computation cost. In addition, if a subset of negative examples is randomly selected from N, negative examples located in certain regions of the feature space may not be selected. As a result, the decision boundary of the SVM may be incorrectly estimated, and the distance between positive and negative examples would be incorrect. Thus, we collect a set of *representative negative examples* that characterize the distribution of all negative examples. To this end, we group negative examples into clusters using the k-means clustering algorithm and the Euclidian distance measure. It should be noted that since various semantic contents are presented in negative examples, their features are very diverse. Hence, a large number of clusters are necessary to capture the large diversity of features in negative examples. Therefore, we use a parameter β to control the number of clusters relative to the number of negative examples (line 2 in Algorithm 1). In our experiment, β is set to 10 so that when $|N| = 30,000$, $3,000$ clusters are obtained.

For each cluster c, the most centrally located negative example is selected as the representative example n_c:

$$n_c = \min_{n_i \in N_c} \sum_{n_j \in N_c} dist(n_i, n_j) \tag{1}$$

where N_c is the set of negative examples in c, n_i and n_j are respectively the i-th and j-th negative examples in N_c, and $dist(n_i, n_j)$ represents their Euclidian distance. Thus, n_c is selected as the negative example having the minimum sum of Euclidian distances to the other negative examples in N_c. A set of representative negative examples for all clusters is

denoted as RU.

Next, an SVM is built on P and RU and check whether or not each negative example n in N is distant from the decision boundary of the SVM, using the following criterion:

$$\left|\frac{w \bullet n - b}{||w||}\right| > \gamma \quad (2)$$

where the left hand side represents the distance between n and the decision boundary. Specifically, $\frac{w \bullet n}{||w||}$ means that n is projected onto w that is the normal vector of the hyperplane, and normalized by the norm of w. $\frac{b}{||w||}$ is the offset of the hyperplane from the origin. The distance between n and the hyperplane is computed as the absolute value of the subtraction between $\frac{w \bullet n}{||w||}$ and $\frac{b}{||w||}$. Since w is comprised of support vectors and their weights, the distance can be computed only using the product of n and each support vector. Thus, although a non-linear SVM projects n into a higher-dimensional feature space, the distance between n and the hyperplane can be computed by the 'kernel trick' where the product of n and a support vector is defined by a kernel function (see [2] for more details). n is filtered if the distance defined in equation 2 is larger than the threshold γ. This filtering is iterated until the number of iterations reaches the maximum number α or when no further negative examples are removed from N. The resulting N only includes negative examples that are similar to P.

B. Concept Detection Using Rough Set Theory

RST requires features that classify positive and negative examples imperfectly. Thus, it is not appropriate to apply RST directly to features described at the beginning of section II. The reason is the high-dimensionality of the bag-of-visual-words representation having 1,000 dimensions. It is likely that positive and negative examples can be perfectly classified by a hyperplane in the high-dimensional feature space. Consequently, RST only extracts a small number of classification rules that characterize the entire set of positive examples. Such rules are only useful for identifying shots, where appearances of a concept and surrounding settings are very similar to the ones in positive examples; these rules are not useful for identifying shots where the concept is present in a variety of appearances and settings. Therefore, for the construction of an SVM, we use a subset of randomly selected positive examples, and a subset of negative examples that are selected by our NES method. In other words, the SVM may incorrectly classify positive and negative examples that were excluded during the SVM construction. Such a SVM is constructed on each feature, and its classification result is used as a feature in RST.

Although the above SVMs characterize different sets of shots, they are not very accurate due to the insufficiency of positive and negative examples. A simple combination of these SVMs like majority voting can result in many false positives. That is, a concept is wrongly regarded to be presented in shots where a concept is absent. To overcome this, we utilize RST to analyze SVM classification results and extract rules as combinations of SVMs. Each rule can correctly discriminate a subset of positive examples from the whole set of negative examples. This enables us to cover a variety of shots where the concept is present, by alleviating false positives. In what follows, we summarize the rule extraction procedure of RST. Please refer to [6] in more detail.

For each pair of a positive example p_i and a negative example n_j, we first determine $SVM_{i,j}$ that is a set of SVMs useful for discriminating p_i from cj. If an SVM can correctly classify p_i and n_j, it is included in $SVM_{i,j}$. In other words, p_i can be discriminated from n_j when at least one SVM in $SVM_{i,j}$ is utilized. Next, we compute the discernibility function df_i, that represents sets of SVMs, required to discriminate p_i from all negative examples. This is achieved by using at least one SVM in $SVM_{i,j}$ for all negative examples. That is, df_i is computed by taking the disjunction of SVMs in $SVM_{i,j}$, and then taking the conjunction of such disjunctions for all negative examples:

$$df_i = \wedge\{\vee SVM_{i,j} |\ 1 \leq j \leq N\}, \quad (3)$$

where N is the number of negative examples. df_i is simplified into the minimal disjunctive normal form, where each conjunctive term, called *reduct*, represents a minimal set of SVMs required to discriminate p_i from all negative examples[1]. Such a reduct forms a rule: "The concept is regarded to be present in a shot, if all SVMs in the reduct classify it as positive". Shots that match many rules comprise the detection result of the concept.

C. Experimental Results

Although we submitted the run *L_A_cs24_kobe_sin_1*, this contains bugs in the parallelization process of matching test shots with rules extracted by RST. In Fig. 2, the left and right bars for each concept represent the average precisions of the run with bugs (*L_A_cs24_kobe_sin_1*) and the bug-fixed run, respectively. As can be seen from Fig. **??**, by fixing bugs, the performance of our run is significantly improved. However, our method has

Fig. 3 shows a performance comparison between the case of using negative examples selected by our NES method and the case of using randomly selected negative examples. For each concept, average precisions of the former and latter cases are represented by the left and right bars, respectively. Fig. 3 illustrates that the overall performance using negative examples selected by our NES method is superior to the one using randomly selected negative examples.

Regarding 23 concepts in TRECVID 2011 SIN task, Fig. 4 shows mean average precisions of our SIN method, SVMs

[1]Although the simplification of df_i is NP-hard, an approximate solution can be obtained using the genetic algorithm [4].

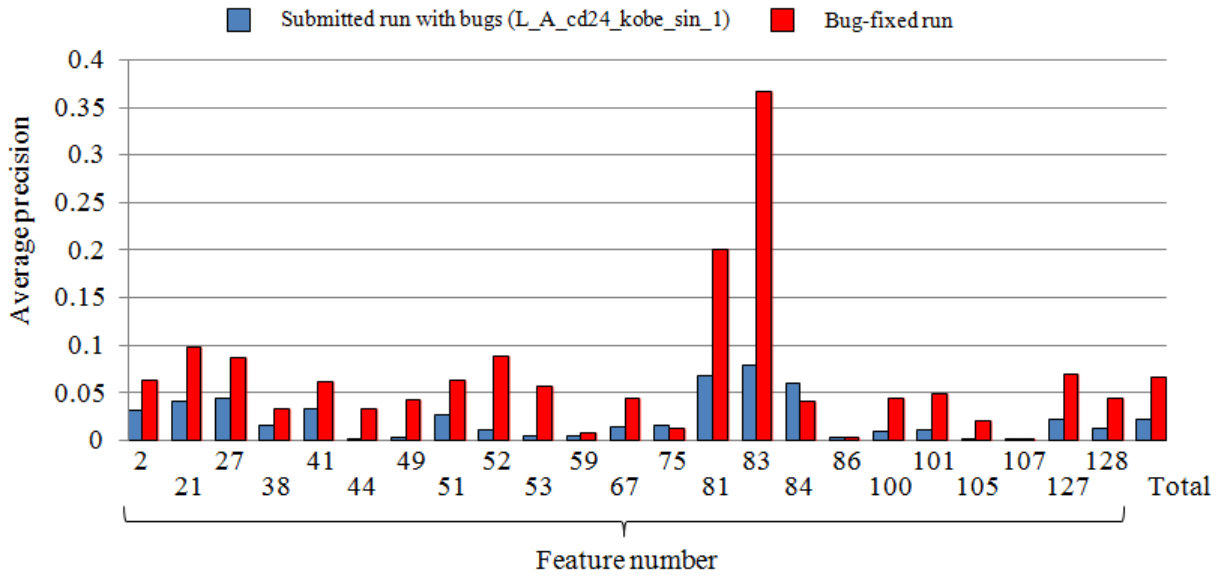

Figure 2. Performance of our SIN method.

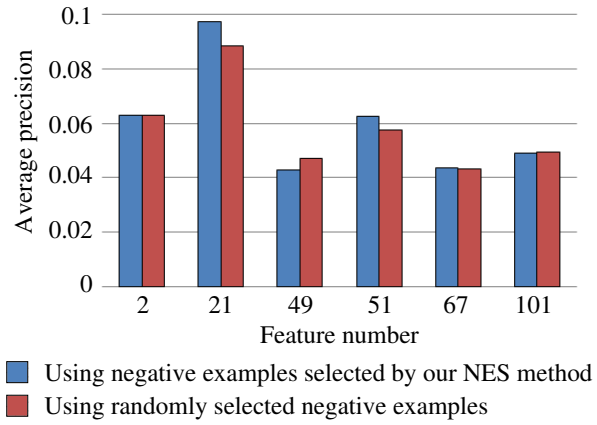

Figure 3. Performance comparison between the case of using negative examples selected by our NES method and the case of using randomly selected negative examples.

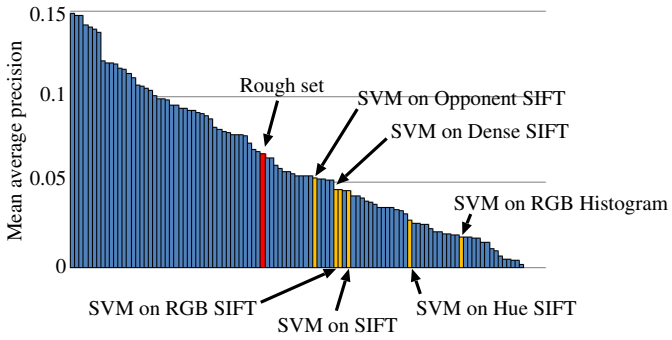

Figure 4. Performance comparison between our SIN method, SVMs constructed for individual features, and methods submitted to TRECVID 2011 SIN task.

constructed for individual features (i.e. these SVMs constitutes rules), and methods submitted to the task. Fig. 4 validates the effectiveness of RST where SVMs are combined into classification rules. However, there is currently a big gap between our SIN method and top-ranked methods. One main reason is that due to the computational complexity, for each concept, we limit the maximum number of positive examples to $1,000$. This is clearly insufficient for accurate concept detection, since more than $10,000$ positive examples are available for some concepts like *Indoor* and *Male_Person*. We are now conducting an additional experiment using all of the available positive examples.

III. Multimedia Event Detection

Our MED method consists of two main processes, *virtual example creation* and *event detection* processes. In the former process, virtual examples for an event are created by synthesizing user's gesture in front of a video camera, a 3D object and background images. In the event detection process, using virtual examples, we construct a classifier that discriminates videos where the event occurs from videos where it does not occur. Below, the above two processes are described in more detail.

A. Virtual Example Creation

Fig. 5 illustrates the virtual example creation process. As can be seen from 5 (a), a virtual example is created by synthesizing user's gesture in front of a video camera, 3D

object and a background image. Fig. 5 (b) shows an overview of our virtual example creation system. To synthesize a 3D object in synchronization of user's gesture, we use a *wireless magnetic sensor system* as shown in the bottom of Fig. 5 (b). In particular, Polhemus *LIBERTY LAUTS* (Large Area Tracking Unthethered System) [7] is used. This can track the position and orientation of a wireless magnetic marker in the 6 degrees of freedom (i.e. x-y-z axes and the rotation on each axis). A 3D object is placed and rotated based on marker's position and orientation, respectively. The reason for using LIBERTY LATUS is its magnetic-sensing ability, which enables us to track marker's position and orientation even if it is occluded by user's body parts. In addition, LIBERTY LATUS includes a System Developer's Kit (SDK) which facilitates the integration of the wireless magnetic marker tracking into one's own system. In our case, Solidray Co., Ltd. and we have jointly extended the virtual 3D space construction tool, *OmegaSpace* [8], to synthesize a 3D object with user's gesture.

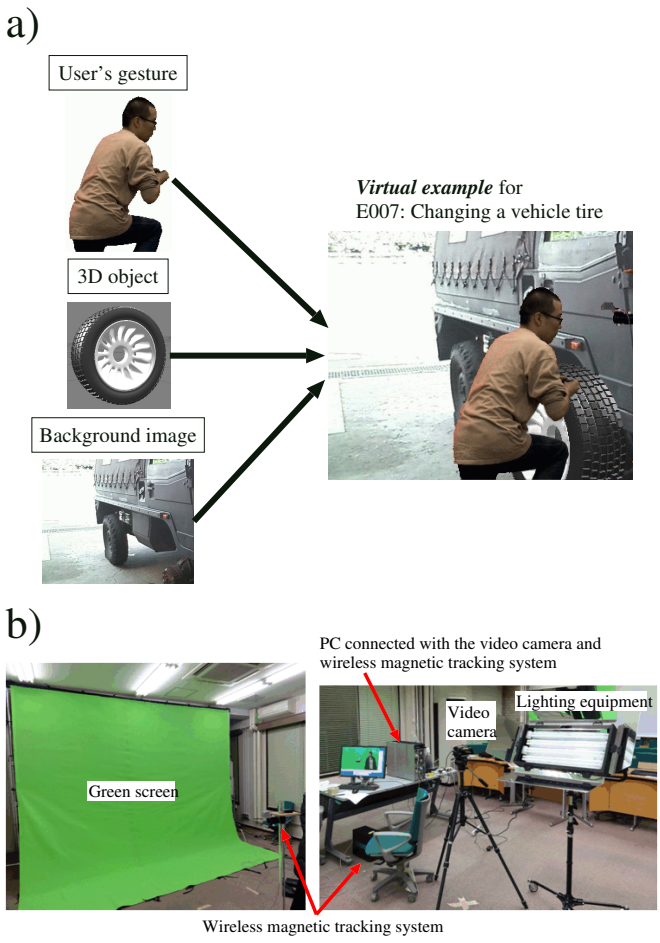

Figure 5. Illustration of virtual example creation.

A background image is synthesized with user's gesture and the 3D object. To this end, we conduct a background subtraction to extract the region of the user and 3D object. Then, the extracted region is synthesized with a background image. As shown in Fig. 5 (b), we adopt *chroma keying technique* using the green screen and lighting equipment. Note that we tried the background image synthesis without using chroma keying technique. However, due to the shadow which emerged during user's gesture, the region of his/hers (and the 3D object) could not be accurately extracted. Thus, we remove the shadow by lighting the user using the green screen and lighting equipment. The extracted region of the user and 3D object is synthesized with a background image, so that a virtual example is created. Note that the same event is taken by different camera techniques and settings, where a person and an object appear in different sizes, positions and orientations on the screen. Hence, we create various virtual examples by synthesizing different users' gestures, 3D objects and background images.

B. Event Detection

First of all, all videos are divided into shots using our own shot detection method, which detects a shot boundary as a significant change in color and edge features between two consecutive frames. Then, from the middle frame in each shot, *Dense SIFT* feature is extracted using the software developed by Sande et al. [5], and represented using a $1,000$-dimensional bag-of-visual-words representation. Based on this shot representation, we develop an event detection method.

Our event detection method focuses that an event consists of several subevents. For example, we assume that the event "E007: Changing a vehicle tire" consists of the following five subevents: "Bringing a new tire", "Putting off an old tire", "Putting on the new tire", "Screwing up nuts" and "Carrying away the old tire". For each of such subevents, an SVM is constructed where virutal examples are used as positive examples. In addition, we manually select poitive examples from training videos. Also, we regard randomly selected shots as negative examples. Each shot in test videos is annotated with the 'discriminant value', that is propotional to the distance between the shot and the decision boundary of the SVM. A large discriminant value indicates a high degree of relevance of a shot to the subevent. After annotating shots in each test video using discriminant values of SVMs for all subevents, we examine whether the event occurs in this test video. Specifically, we identify which shot is the most relevant to each subevent by finding the one-to-one matching between shots and subevents using Kuhn-Munkres algorithm [9]. The final evaluation value of the test video is the sum of discriminant values of shots, each of which is matched with a subevent.

C. Experimental Results

In order to examine the effectivenss of virtual examples, we compare the following two methods: One method is *c_real* where the SVM for each subevent is constructed only using positive examples selected from training videos. The other method is *p_virtual* which uses both virtual examples and positive examples selected from training videos. Note that *p_virtual* and *c_real* use the same set of negative examples that are shots randomly selected from training videos. Fig. 6 shows detection error tradeoff curves of *p_virtual* and *c_real*. As can be seen from this figure, the detection error curve of *p_virtual* is closer to the origin than that of *c_real*, that is, the former is superior to the latter. This indicates the effectiveness of virtual examples.

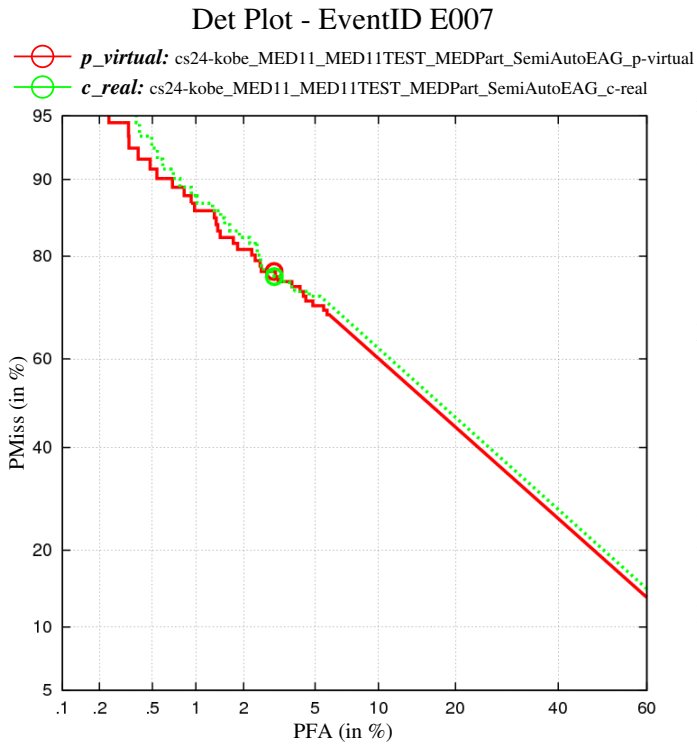

Figure 6. Performance comparison between *p_virtual* and *c_real*.

We use Fig. 7 to closely investigate the effectiveness of virtual examples. Fig. 7 shows detection error curves of *p_virtual*'s and *c_real*'s SVMs for five subevents. As shown in Fig. 7, for *Subevent 1*, *4* and *5*, SVMs of *p_virtual* are superior to those of *c_real*. For *Subevent 3*, their SVMs are comparable. For *Subevent 2*, the SVM of *p_virtual* is outperformed by that of *c_real*. It can be seen that using virtual examples is effective especially in a case where only a small number of positive examples are available. For example, we can only select five positive examples for *Subevent 1* and *5*, and adding virtual examples significantly improve the detection performance of these subevents. In comparison, for *Subevent 3* where 156 positive examples are available, little improvement is obtained by adding virtual examples. The reason why the performance is degraded by adding virtual examples for *Subevent 2*, is that virtual examples are too unnatural. In the future, we will investigate the effectiveness of virtual examples for much more variety of events.

IV. Conclusion and Future Works

In this paper, we presented our methods and experimental results for TRECVID 2011 SIN and MED tasks. For SIN task, considering the expensive computational cost of an SVM, we developed a method that selects a small number of negative examples similar to positive examples. Such negative examples are useful for characterizing the decision boundary between shots where a concept is present and shots where it is absent. Additionally, in order to cover a large variety of shots where a concept is present, we used rough set theory to extract multiple classification rules, that characterize different subsets of positive examples. Although the evaluation result of our run was not very good, we found that one main reason is overfitting in RST where extracted rules are very specific to positive examples, and are not effective for shots in test videos. For MED task, considering the difficulty of collecting a sufficient number of positive examples for an event, we developed a method that creates virtual examples by synthesizing user's gesture, 3D object and background images. Evaluation results indicates the effectiveness of virtual examples.

Our future works are as follows: For SIN task, in order to avoid extracting overfit classification rules, we plan to extend the current RST to *variable precision RST* [10]. This enables us to extract rules that discriminate subsets of positive examples from negative examples within an acceptable error. For MED task, in order to validate the generality of our method using virtual examples, it needs to be examined on many events other than the event "E007: Changing a vehicle tire". In addition, although object movements are valuable cues for event detection, our current method only uses an image feature (i.e. *Dense SIFT*). Thus, we aim to incorporate a temporal feature such as *3DSIFT* [11]. Since the extraction of such temporal features seems to require an expensive computation cost, we plan to parallelize the extraction process using tens or hundreds of processors.

References

[1] Ayache S. and Quénot G., "Video corpus annotation using active learning," in *Proc. of ECIR 2008*, 2008, pp. 187–198.

[2] Vapnik V., *Statistical Learning Theory*. Wiley-Interscience, 1998.

[3] Tsang I., Kwok J. and Cheung P., "Core vector machines: Fast svm training on very large data sets," *Journal of Machine Learning Research*, vol. 6, pp. 363–392, 2005.

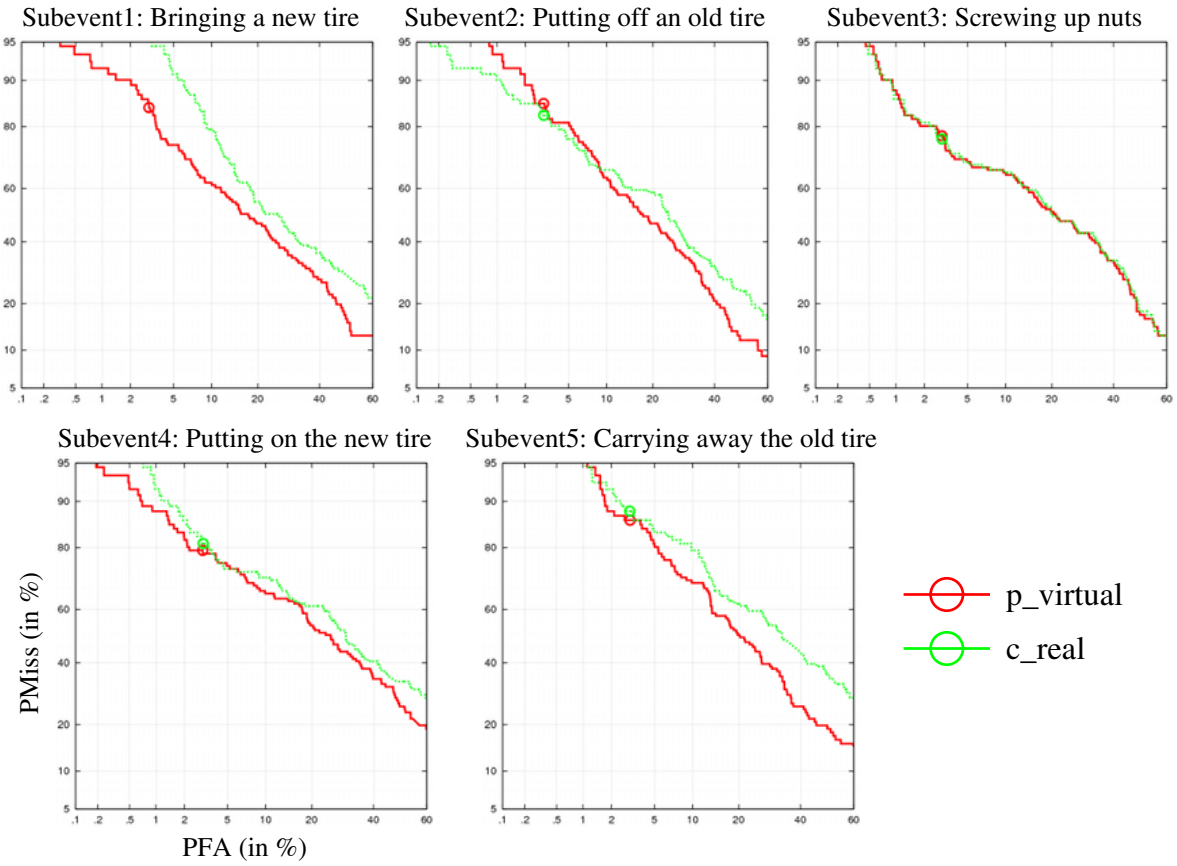

Figure 7. Performance comparison in between p_virtual and c_real for subevent detection.

[4] Komorowski J., Øhrn A. and Skowron A., "The rosetta rough set software system," in *Handbook of Data Mining and Knowledge Discovery, W. Klösgen and J. Zytkow (eds.)*. Oxford University Press, 2002, ch. D.2.3.

[5] Sande K., Gevers T. and Snoek C., "Evaluating color descriptors for object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1582–1596, 2010.

[6] Shirahama K., Matsuoka Y. and Uehara K., "Event retrieval in video archives using rough set theory and partially supervised learning," *Multimedia Tools and Applications*, (in press).

[7] POLEHEMUS, *LIBERTY LATUS wireless motion tracking system from Polhemus*, http://polhemus.com/?page=Motion_Liberty_Latus.

[8] Solidray Co., Ltd., *3D/VR space construction and experience tool Omega Space*, http://www.solidray.co.jp/product/omega/index.html#1 (In Japanese).

[9] Peng. Y. and Ngo C., "Clip-based similarity measure for query-dependent clip retrieval and video summarization," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 612–627, 2006.

[10] Ziarko W., "Variable precision rough set model," *Journal of Computer and System Science*, vol. 46, no. 1, pp. 39–59, 1993.

[11] Scovanner P., Ali S. and Shah M., "A 3-dimensional sift descriptor and its application to action recognition," in *Proc. of ACM MM 2007*, 2007, pp. 357–360.